Reviewer 1:

I am reviewer 1 and these are my comments regarding the paper. This paper presents a new Nearest Neighbor-search algorithm, that reaches the peak performance on Tensor Processing Units or TPUs. The paper is based on observations of hardware architectural properties and the so called roofline performance model. The algorithm is implemented for TPUs as well. The evaluation is done on two TPU versions using two K-Nearest Neighbor datasets, and compared to several Graphics Processing Unit or GPU algorithms.

The strengths of the paper include a nice connection between theoretical observations and practical results, the paper also has good performance and finally the work can have significant practical impact.

However, the weakness stem from limited evaluation as the paper is only evaluated on two versions on TPUs, and two datasets. It Would have been interesting to see how general the algorithm is, for example, would it reach the same performance limits in other hardware platforms also? Finally, the algorithm 1 and 2 descriptions are relatively clear and straight-forward to implement on other platforms. Can the authors elaborate a bit on why it would be such a substantial effort to do? My score for the paper is a weak acceptance.

Reviewer 2:

I am reviewer number 2 and this my review for the paper. It presents an Artificial Neural Network algorithm on TPU and analyzes the memory and instruction bandwidth of the ANN algorithms.

The strengths of the paper are firstly, the problem is well-motivated, secondly, the related work is comprehensively studied and discussed. And the entire story is easy to follow for readers. Thirdly, the methodology to study the algorithm from the hardware bottlenecks is interesting.

The weaknesses are the following:

1. I suggest the authors to include a brief discussion of TPU, for example, what can be done efficiently and what can not be done.

2. The experimental evaluation is a bit problematic. How about other methods on GPUs/TPUs, for example hashing and graph-based methods besides the FAISS baseline?

3. In Figure 3, the highest recall shown for Glove is 0.9. Is there a recall limitation for the proposed algorithm?

4. The technical contribution of the bi-stage partial reduction and scoring is limited.

In addition, I have one suggestion,

1. Is it possible to show that the proposed algorithm can achieve high recalls for most ANN datasets? I suggest the authors to discuss the limitation of the algorithm, for example how to adapt the problem to other common similarity measures. My suggestion is a weak acceptance of the paper.

Reviewer 3:

I am reviewer number 3 and here are my comments for the paper. This paper presents a new algorithm implementation of K nearest neighbor search that is able to increase the arithmetic intensity. The distances between all queries and entries in the database are calculated in L3 BLAS fashion. Then only the top-1 entry is kept in each bin. The bin size can be adjusted based on the requirement of the recall. The roofline analysis shows a better peak throughput while the end-to-end evaluation demonstrates a better throughput-recall trade-off than the previous solutions.

The strengths of the paper are the following:

1. This paper presents a detailed analytical model of the kernel implementation of the BLAS-based operation. The authors conclude the impact of COPs and introduce the algorithm accordingly.

2. The proposed solution is simple yet effective. The adjustment of the bin size provides a simple method to balance recall and throughput.

Next, I believe the paper has the following weaknesses:

1. Unclear hardware motivation. The paper is entitled 'TPU-KNN'. However, throughout this paper, I'm not able to find any information indicating that the proposed algorithm requires any special architectural support from TPU rather than other accelerators. It seems that the same analysis could also apply to GPU or other accelerators.

2. Vague definition of coefficient-wise operations or COPs. I cannot understand the usage of the concept of coefficient-wise operations. In Table 1, comparing with the datasheet of A100 or V100, I can understand COPs as a Floating Point Operation or FLOP in the vector cores. However, in the later description in Algorithm 1, one COP seems to be a general non-matrix operation without considering the problem size. In this case, one COP, for example, could include multiple FLOPs and seems to be unmatched by what is presented in Table 1.

3. Incomplete Evaluation. Based on what I've mentioned as the first weakness, the evaluation part lacks a fair baseline. I can understand the advantage of recall of the proposed algorithm. However, it is hard to understand whether the performance gain over GPU comes from the algorithm itself or the higher efficiency of TPU. To isolate this factor, the authors should either provide the result of the proposed algorithm implemented on GPUs or the baseline algorithms implemented on TPU.

My score for the paper is a rejection.