POSTER: A Memory-Access-Efficient Adaptive Implementation of kNN on FPGA through HLS

Xiaojia Song
Computational Science
San Diego State University
San Diego, CA, USA
xsong2@sdsu.edu

Tao Xie
Computer Science
San Diego State University
San Diego, CA, USA
txie@sdsu.edu

Stephen Fischer
Memory Solutions Lab
Samsung Semiconductor, Inc.
San Jose, CA, USA
sg.fischer@samsung.com

***Abstract*—Implementing an efficient k-Nearest Neighbors (kNN) algorithm on FPGA is becoming challenging due to the fact that both the size and dimensionality of datasets that kNN is working on have been rapidly growing, which may incur a performance bottleneck on the memory-access. To reduce the impact of the memory-access constraint, in this paper we implement two kNN kernels through high-level synthesis (HLS) on FPGA by employing two data access reduction methods: low-precision data representation and principal component analysis based filtering (PCAF). One kernel is called MBFS-kNN (Memory-efficient Brute-Force Searching kNN) and the other is called MPCAF-kNN (Memory-efficient PCAF kNN). Both kernels have been highly optimized to fully exploit the characteristics of FPGA. Besides, they are adaptive to the number of dimensions (D), number of data points in a database (N), number of nearest neighbors (k), number of bits per feature (B), and number of principal components (d). We evaluate the two kernels by comparing them with two state-of-the-art kNN implementations on a high-end CPU server, an existing BFS-kNN kernel on FPGA, and an existing BFS-kNN kernel on GPU. Our results show that the external memory-accesses of these two kernels are greatly reduced and our design outperforms the existing ones.**

***Keywords*-kNN, FPGA, High-level synthesis, Low-precision data representation, PCA-based filtering, Memory-access-efficient, Adaptive kernel.**

I. Introduction

The k-Nearest Neighbors (kNN) algorithm is one of the most popular machine learning algorithms and has been applied in a wide range of HPC applications such as image/video retrieval , big data analysis, machine learning, and computer vision [4] [12]. Meanwhile, an FPGA-based heterogeneous system is becoming increasingly attractive for the HPC. For example, Microsoft is employing FPGAs to accelerate its Bing page ranking functions [2]. Baidu developed a software-defined accelerator for large-scale deep neural network systems, which heavily rely on FPGA devices [9].

The accelerations of kNN using FPGA in previous work has demonstrated a pretty inspiring results [5] [10]. However, a challenge is emerging due to the fact that both the size and dimensionality of datasets that kNN is working on have been rapidly growing these days. For example, the number of images in Tineye's indexed image database has increased from 0.7 billion in 2008 to 35 billion in 2019 [14]. At the same time, to obtain a more accurate representation of an image, the number of dimensions of each feature vector extracted by some neural network technology could be as large as 4,096 [11]. As a result, kNN searching in such a large database with a high dimensionality becomes both compute-intensive and memory-intensive [7]. Before the power of internal high parallelism and deep pipeline of the FPGA can be leveraged, the external memory access bottleneck badly needs to be removed.

To reduce the impact of the memory access constraint, in this paper we implement two kNN kernels through high-level synthesis (HLS) [13] on FPGA by employing two data access reduction methods: low-precision data representation [3] and principal component analysis based filtering (PCAF) [4]. Low-precision data representation has been successfully applied in various domains as it can improve hardware bandwidth utilization by lowering data precision, and thus, reducing the volume of data being read/written [3] [6]. PCAF, on the other hand, uses a data filtering mechanism to exclude those reference features that are not likely to be k-NN features according to the PCA estimation [4]. One of the kernels we implemented is called MBFS-kNN (Memory-efficient Brute-Force Searching kNN) and the other is called MPCAF-kNN (Memory-efficient PCAF kNN). While the former only employs the approach of low-precision data representation to reduce memory access, the latter utilizes both methods to achieve the same goal. MBFS-kNN can be used to carry out an accurate kNN search, whereas MPCAF-kNN can only perform an approximate kNN search. Although the idea of PCAF is borrowed from a recent research work [4], this study is the first attempt to apply PCAF in kNN kernel implementation on FPGA. Both kernels have been highly optimized to fully exploit the characteristics of FPGA. Besides, they are adaptive to the number of dimensions (D), number of data points in a database (N), number of nearest neighbors (k), number of bits per feature (B), and number of principal components (d for MPCAF-kNN).

978-1-7281-3613-4/19/$31.00 ©2019 IEEE
DOI 10.1109/PACT.2019.00066

II. Evaluation

We evaluate the two kNN kernels in terms of performance and energy-efficiency by comparing them with two state-of-the-art kNN implementations on a high-end CPU server, an existing BFS-kNN kernel on FPGA, and an existing BFS-kNN kernel on GPU. Two datasets are used.

1) Datasets: KDD-CUP consists of 50,000 data points and each point has 64 features [1]. GIST1M is 3.8 GB and contains one million 960-dimensional data points extracted from a variety of images by using global color GIST descriptors [8].

2) Platforms: ***CPU server:*** PowerEdge R730xd Rack Server has two Intel(R) Xeon(R) CPU E5-2699 @ 2.20GHz. Each CPU has 22 physical cores and 2 threads can run on each core (i.e., totally 88 threads). The server has 128 GB DDR4. ***FPGA platform in our work:*** VCU1525 [13]. There are 4 SDRAM banks available. The maximal bandwidth to access an individual memory bank is 512 bits per clock cycle. ***FPGA platform in [10]:*** The FPGA board in [10] is a Terasic DE4 with a Stratix IV 4SGX530 FPGA and two DDR2 memory banks. The maximal bandwidth of one memory bank is 12.75 GB/s. ***GPU platform in [10]:*** The GPU used in [10] is an AMD Radeon HD7950 with 28 compute units(900 MHz). The board consists of a 3 GB GDDR5 memory with 240 GB/s bandwidth.

The experimental results demonstrate that MBFS-kNN can achieve a performance equivalent to that of a 76-thread CPU server in the best case. It also outperforms the two existing BFS-kNN kernels in execution time and energy-efficiency by 5.5x and 1.97x, 7.45x and 22.23x, respectively. The MPCAF-kNN kernel achieves up to a performance equivalent to that of a 56-thread of CPU server. It also gains 324x energy-efficiency compared with the CPU server. Compared with the BFS-kNN, MPCAF-kNN reduces external memory accesses by 28~231x. This paper makes the following contributions. First, to the best of our knowledge, this is the first research utilizes a PCA-based data filtering mechanism to reduce memory accesses of a kNN on FPGA. Second, we apply optimization on MPCAF-kNN for performance scalability, which applies not only to the implementation on FPGA but also on CPU or GPU. Third, a comprehensive evaluation of the two kernels in performance and energy-efficiency is provided.

III. Conclusions

In this paper we design and implement two kNN kernels on FPGA through HLS. The implementations have been highly optimized to fully exploit the characteristics of FPGA. Two data access reduction methods (i.e., low-precision data representation and PCAF) are employed to reduce the number of external memory accesses. The two kernels are adaptive to all key parameters. Further, we evaluate them with different settings. The experimental results show that our optimized kNN kernels outperform existing ones in both execution time and energy-efficiency. We plan to release the source code of the two kernels to benefit the community.

References

[1] Kdd-cup 2004 quantum physics data set. https://www.kdd.org/kdd-cup/view/kdd-cup-2004/Data, 2004.

[2] Microsoft extends fpga reach from bing to deep learning. https://www.nextplatform.com/2015/08/27/microsoft-extends-fpga-reach-from-bing-to-deep-learning/s, 2015.

[3] J. Eilert, A. Ehliar, and D. Liu. Using low precision floating point numbers to reduce memory cost for mp3 decoding. In *6th Workshop on MSP, 2004.*, pages 119–122.

[4] H. Feng, D. Eyers, S. Mills, Y. Wu, and Z. Huang. Principal component analysis based filtering for scalable, high precision k-nn search. *IEEE TOC*, 67(2):252–267, 2018.

[5] H. M. Hussain, K. Benkrid, and H. Seker. An adaptive implementation of a dynamically reconfigurable k-nearest neighbour classifier on fpga. In *2012 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, pages 205–212.

[6] K. Kara, D. Alistarh, G. Alonso, O. Mutlu, and C. Zhang. Fpga-accelerated dense linear machine learning: A precision-convergence trade-off. In *2017 IEEE 25th Annual International Symposium on FCCM*, pages 160–167.

[7] V. T. Lee, A. Mazumdar, C. C. del Mundo, A. Alaghi, L. Ceze, and M. Oskin. Application codesign of near-data processing for similarity search. In *2018 IEEE IPDPS*, pages 896–907.

[8] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[9] J. Ouyang, S. Lin, W. Qi, Y. Wang, B. Yu, and S. Jiang. Sda: Software-defined accelerator for large-scale dnn systems. In *2014 IEEE Hot Chips 26 Symposium (HCS)*, pages 1–23.

[10] Y. Pu, J. Peng, L. Huang, and J. Chen. An efficient knn algorithm implemented on fpga based heterogeneous computing system using opencl. In *2015 IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines*, pages 167–170. IEEE, 2015.

[11] A. Shah, R. Naseem, S. Iqbal, M. A. Shah, et al. Improving cbir accuracy using convolutional neural network for feature extraction. In *2017 13th International Conference on Emerging Technologies (ICET)*, pages 1–5. IEEE, 2017.

[12] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling features for large scale partial-duplicate web image search. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 25–32. IEEE, 2009.

[13] Xilinx. Xilinx virtex ultrascale+ fpga vcu1525 acceleration development kit.

[14] W. Zhou, H. Li, and Q. Tian. Recent advance in content-based image retrieval: A literature survey. *arXiv preprint arXiv:1706.06064*, 2017.