The Implementation of a KNN Classifier on FPGA with a Parallel and Pipelined Architecture based on Predetermined Range Search

Miren Tian, Xin'an Wang*, Xing Zhang, Zhiqiang Yang, Jipan Huang, Hao Chen

The Key Laboratory of Integrated Microsystems
Peking University Shenzhen Graduated School, Shenzhen 518055, China
* Email: anxinwang@pku.edu.cn

Abstract

K-nearest neighbor (KNN) classification algorithm performs slowly for large scale training set and high dimensions. To overcome the disadvantage, we need to focus on the points within a predetermined range, without changing the precision. This method is named Predetermined Range Search (PRS). In this paper, we proposed a method to find the reference distance (ReDist), a parallel and pipelined architecture based on PRS to implement KNN classification algorithm on FPGA. Besides, we use real SPECT dataset for evaluation. The result shows that clock frequency is up to 186.4MHz on Virtex 4 which is 1.4x faster than the conventional design. Meanwhile, this novel architecture has a smaller BRAMs (Block RAMs) coverage and a simpler circuit structure.

1. Introduction

K-nearest neighbor (KNN) classification algorithm, one of the easiest classification methods of data mining, is mainly used for bioinformatics, medical image processing, data retrieval, pattern recognition and machine-learning workloads etc. [1-3].

This classification algorithm can achieve a high classification of accuracy in problems of unknown distributions, compared with those commonly used parametric classification approaches, such as linear classifiers and quadratic classifiers [4]. However, test vector to be classified must calculate its distance from all the training vectors in order to obtain its K-nearest neighbor points, resulting in high computational complexity and large time delay. In order to overcome the slow speed of classification for large scale training set and high dimensions, we just need to focus on the points within a predetermined range, without changing the precision. Therefor we proposed a method to find the ReDist. Besides, we design a parallel and pipelined architectures based on PRS to implement KNN classification algorithm on FPGA.

2. Previous works

As is reported in paper [5], if a testing vector with M dimensions is known as $Y = \{y_1, y_2, y_3, \cdots, y_M\}$, and N training vectors each known as $X_{NM} = \{x_{N1}, x_{N2}, x_{N3}, \cdots, x_{NM}\}$, then the Euclidian distance between the testing vector Y and one training vector X is:

$$DistE(X,Y) = \sqrt{\sum_{i=1}^{M}(x_{Ni} - y_i)^2} \quad (1)$$

Although Euclidian distance is widely used in data mining to measure the absolute distance between vectors in a multidimensional space, it is more complex than Manhattan distance in terms of hardware implementation on FPGA [5]. The Manhattan distance between the testing vector Y and one training vector X is:

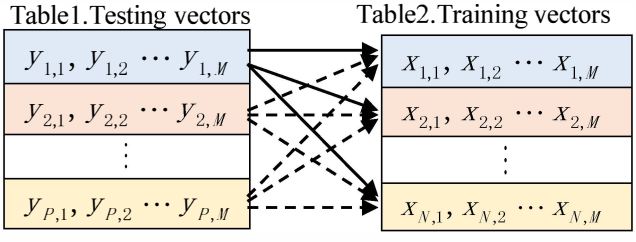
$$DistM(X,Y) = \sum_{i=1}^{M}(|x_{Ni} - y_i|) \quad (2)$$

Most related works on hardware implementation of the KNN classification algorithm were reported in [1] and [3], including the SIMD-style parallel architecture [6], the linear systolic array [7], wavelet transform and partial distance search (PDS) [4] etc. In addition, vector reconfigurable K-nearest neighbor accelerator with adaptive precision is presented in [2], which can find the nearest neighbor out of 128×128-D randomly generated vectors for 21.5M testing vectors/s at 16 cycles/vector with Manhattan distance measured at 338MHz, nominal 750mV, 25°C. However, it can't apply to any vectors with different dimensions.

3. Architecture design

An optimized KNN classification algorithm is proposed in this work. Meanwhile, we design a parallel and pipelined architecture based on PRS.

To narrow the search, we propose a method to find the ReDist, which is applicable to the dataset with the number of training vectors (N) with M dimensions, testing vectors (P) with M dimensions, categories (C), K nearest neighbours (K), and number of bits per dimension (B). This method is described below.



978-1-4673-9719-3/16/$31.00 ©2016 IEEE

Table 3. Manhattan distance

$z_{1,1}$	$z_{1,2}$	$\cdots$	$z_{1,P}$
$z_{2,1}$	$z_{2,2}$	$\cdots$	$z_{2,P}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$z_{N,1}$	$z_{N,2}$	$\cdots$	$z_{N,P}$

Table 4.The number of valid data

Num	V_1	V_2	$\cdots$	V_p

The numerical relationship between table 1, table2 and table 3 can be shown by the following formula.

$$z_{n,p} = \sum_{i=1}^{M} | y_{pi} - x_{ni} | \qquad (3)$$

If $z_{n,p} \leq \mathrm{Re}\,Dist$, $z_{n,p}$ is valid. In table 3, $V_1, V_2, V_3, \cdots, V_P$ is the number of valid data in column 1, column 2, column 3 … column p respectively. $V_{\min}$ and $V_{\max}$ are the minimum and maximum of $\{V_1, V_2, V_3, \cdots, V_P\}$ respectively, The ReDist should be smallest possible if $V_{\min} \geq K \geq C$ satisfied. By this way, we can find the most appropriate ReDist.

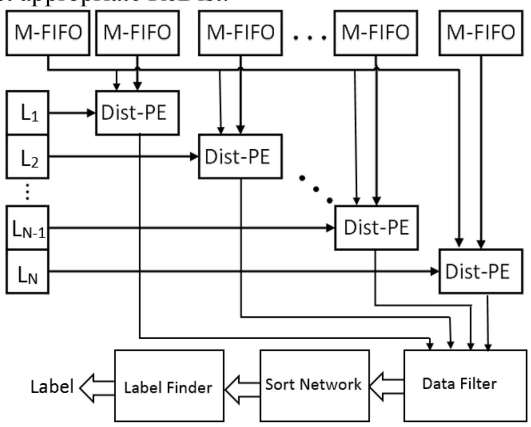


Figure 1. The architecture of the KNN classification algorithm

The architecture of the KNN classification algorithm is shown in figure 1. As is shown, the architecture has been divided into four modules to perform the four main kernels within the KNN classifier which are: the Distance Calculation, Data Filter, Sort Network, Label Finder. More details about them will be presented below.

3.1 Distance Calculation and Data Filter

The Distance Calculation module consists of N distance processing elements (Dist-PEs), N+1 M-FIFOs, and a L-width N-depth RAM which store N labels $(L_1, L_2, L_3, \cdots, L_N)$ corresponding to N training vectors respectively. Each of these FIFOs has a depth of M and a width of B-bit, one of them is to store a testing vector, the other N FIFOs are to store N training vectors respectively. The architecture of the Dist-PE is shown in figure 2.

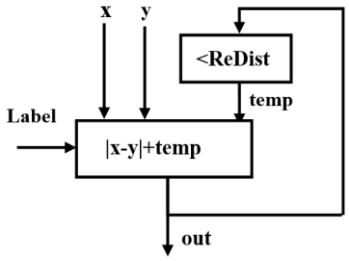


Figure 2. The architecture of the Dist-PE

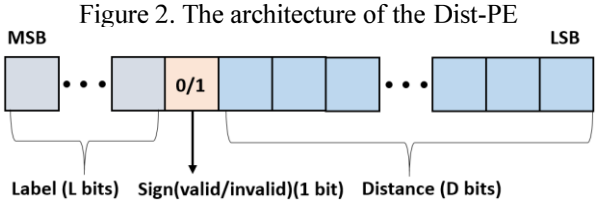


Figure 3. The output format

The output format is shown in figure 3, the Label bits are determined by the value of C, while the Distance bits are determined by the value of ReDist. Sign bit determines the validity of the output, if sign=1, the output is invalid; in contrast, the output is valid. The partial distance between the testing vector Y and one training vector X is given by (4).

$$PDM(X,Y) = \sum_{i=1}^{a} (| x_{Yi} - y_i |) \qquad (4)$$

If $a < M$ and $PDM(X,Y) > \mathrm{Re}\,Dist$, let sign=1, if $a = M$ and $PDM(X,Y) \leq \mathrm{Re}\,Dist$, let sign=0.

The main role of the Data Filter module is to filter out invalid data, and send valid data to the next block sequentially.

3.2 Sort Network and Label Finder

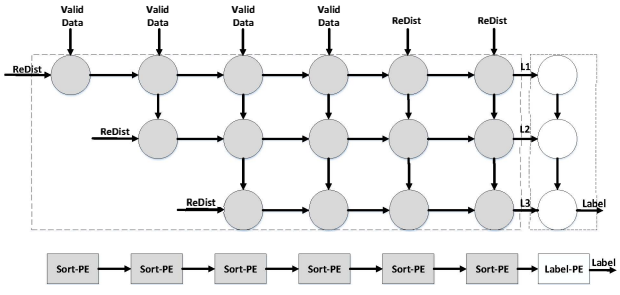


Figure 4. The Sort Network ($V_{\max}$=6, K=3, the actual number of valid data V_{act}=4.)

The two modules are shown in figure 4, which are based on the architectures presented in [1] and [3]. The Sort

Network has $K \times V_{\max} - (K-1)K/2$ nodes that find the K nearest neighbors, and the Label Finder has K nodes that find the majority of the KNN labels to be assigned to the input testing vector.

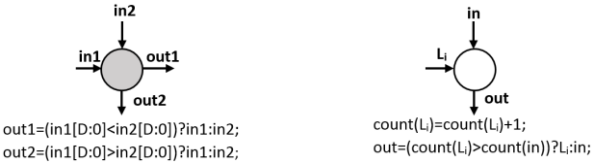


Figure 5. The functionality of each of the PEs shown in figure 4

Receiving the first valid data, the top note of the Sort Network which is directly connected to the Data Filter becomes active sequentially. If V_{act} is less than $V_{\max}$, the upper-right part of ($V_{\max}$-V_{act}) "light grey" notes would receive the ReDist rather than valid data. Every "white" note is mainly composed of C counters to calculate the number of each category.

4. Implementation results

Since the proposed KNN classifier architecture has been improved based on the architectures (A1, A2) presented in [1] and [3], it is compared with A1 and A2 in terms of the number of clock cycles, Dist-PEs, Sort-PEs, Label-PEs and FIFOs to store training set. The comparison is shown in table 5.

Table 5. Comparison

	Clock cycles	Dist -PE	Sort -PE	Label -PE	FIFO
A1	N+M+K	M	K	1	M N-FIFOs
A2	N+M+K	N	N	1	N M-FIFOs
This architecture	$V_{\max}$+M +K	N	$V_{\max}$	1	N M-FIFOs

Due to $V_{\max}$<N, this architecture has a faster processing speed than A1 or A2. In addition, it has simpler circuit structure than A2. This architecture achieves a comparable memory requirement to store training vectors compared with A1 or A2. But in this architecture, the Distance bits are determined by the value of ReDist, while in A1 or A2, they are determined by the value of the maximum in table 3, overall this architecture has a smaller BRAMs (Block RAMs) coverage.

For the validation we used a publically available dataset on cardiac Single Proton Emission Computed Tomography (SPECT) images with N=80, P=187, M=44, C=2 [8][9]. To guarantee the precision of the KNN classification algorithm, simplify the circuit configuration and save storage resources, when set K=5, we can get ReDist=255, $V_{\min}$=5, $V_{\max}$=36, B=8, L=1, D=8 by adopting the method employed in the previous section. The design has been implemented on Virtex 4.

Table 6. Achieved performance for SPECT (FPGA: xc4vfx12-12sf363)

	Clock Frequency (MHz)	Slices Actual/ Total	Slices FF Actual/ Total	BRAMs Actual/ Total
A1	132.6	995/ 5472	780/ 10944	6/36
A2	140.8	2376/ 5472	3186/ 10944	6/36
This architecture	186.4	2464/ 5472	2560/ 10944	4/36

The results, compared with A1 and A2, show that the clock frequency is up to 186.4MHz on Virtex 4 which is 1.4x faster than A1, 1.26x faster than A2, meanwhile, BRAMs coverage is reduced by 1/3.

5. Conclusion

In this paper, we proposed a method to find the ReDist and a parallel and pipelined architecture based on PRS to implement KNN classification algorithm on FPGA. The implementation of the novel architecture on Virtex 4 shows that the clock frequency is up to 186.4MHz which is 1.4x faster than A1, 1.26x faster than A2, meanwhile, BRAMs coverage is reduced by 1/3. And the proposed design has a more simple circuit structure.

Acknowledgments

This work is supported by the Shenzhen Science and Technology Program Fundamental Research Key Project (NO.JCYJ20140417144423206).

References

[1] Manolakos E S, Stamoulias I. IEEE International Symposium on Circuits and Systems. IEEE, p.4133-4136(2010).

[2] Kaul H, Anders M A, Mathew S K, et al. IEEE International Solid-State Circuits Conference. IEEE, (2016).

[3] Manolakos E S, Stamoulias I. IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum. IEEE, p.1-4(2010).

[4] Li H Y, Yeh Y J, Hwang W J. Image Analysis & Recognition, 4633:p.1105-1116(1970).

[5] Hussain H M, Benkrid K, Seker H. Adaptive Hardware and Systems. p.671-678(2012).

[6] Lucas S M. 3:1867-1872 vol.3(1998).

[7] Chen Y A, Lin Y L, Chang L W. IEEE Transactions on Computers, 41(1):p.103-108(1992).

[8] Kurgan L A, Cios K J, Tadeusiewicz R, et al. Artificial Intelligence in Medicine, 23(2):149-169(2001).

[9] UCI Machine Learning Data Repository, http://archive.ics.uci.edu/ml/datasets/SPECTF+Heart