Improve Image Classification by Convolutional Network on Cambricon

Peng He[1,2*], Ge Chen[1,2*], Kai Deng[1,2], Ping Yao[1], and Li Fu[1]

[1] Institute of Computing Technology Chinese Academy of Sciences, HaiDian BeiJing 110108, China
[2] University of Chinese Academy of Sciences, Shijingshan BeiJing 14430, China
{hepeng18s,chenge18s,dengkai19s,yaoping,fuli}@ict.ac.cn

Abstract. Cambricon provides us with a complete intelligent application system, how to use this system for deep learning algorithms development is a challenging issue. In this paper, we exploit, evaluate and validate the performance of the ResNet101 image classification network on Cambricon with Cambricon Caffe framework, demonstrating the availability and ease of use of this system. Experiments with various operational modes and the processes of model inference show, the optimal running time of a common ResNet101 network that classifies the CIFAR-10 dataset on Cambricon is nearly three times faster than the baseline. We hope that this work will provide a simple baseline for further exploration of the performance of convolutional neural network on Cambricon.

Keywords: Cambricon · Convolutional Neural Network · ResNet101.

1 Introduction

Image classification is a fundamental task in the field of computer vision that labels a picture to distinguish different categories visually and concisely. Classification task can also guide the development of other tasks: object detection [11], segmentation [15] and many more, such as instance segmentation, which performs per-pixel labeling of pictures at instance level. Therefore, since ResNet [17] was proposed as an effective image classification network at 2015, superior performance has enabled it to be applied as a backbone network to almost all other tasks [11, 10, 37]. Whether the ResNet network can run normally should be regarded as the basic standard to test whether an intelligent application system can run robustly.

As the calculation of neural network is a quite computationally intensive process, the computing capability of traditional CPU is far from meeting current computational complexity. Even the GPU is not designed originally specifically for artificial intelligence algorithms. The Cambricon chip is the first deep learning processor in the world as far as we know. It uses the hardware's digital logic structure, NFU (Neural Functional Units), to simulate the neural network

* Equal contribution.

connection structure to execute multiplication, addition, activation and other operations [3]. With ASIC (Application Specific Integrated Circuit) mode, which reduces a lot of unnecessary logic functions [7], Cambricon is extremely fast and consumes very low power, making it a superior alternative to GPU in video parsing, autonomous driving, and many more other real-world scenarios.

This work is implementing the ResNet101 classification network based on Cambricon with CIFAR-10 dataset [22]. By trying various operational modes, we find out the optimal operation strategy and running time to verify the performance and efficiency of Cambricon.

This year, joint with Cambricon, China RISC-V alliance, Sugon, and Intellifusion, BenchCouncil[9, 8, 21, 13, 30] organizes four challenge tracks, including international AI system challenge based on RISC-V[19], international AI system challenge based on Cambricon chip[24, 25, 33], international AI system challenge basced on X86 platform[2, 14, 5] and international 3D face recognition algorithm challenge[34, 12]. The source code of AIBench is publicly available from `http://www.benchcouncil.org/benchhub/AIBench/`(Sign up to get access). Notably, we won the third prize on the Cambricon Track of BenchCouncil Challenges.

2 Related Work

Image Classification. Image classification, a fundamental problem in computer vision, can be described as categorizing images into one of several predefined classes [23]. It forms the basis for other computer vision tasks such as localization [38], detection [11, 10, 16, 26, 27], and segmentation [31, 15, 37]. Traditionally, handcrafted features can be extracted from images using feature descriptors for the purpose of classification. The major disadvantage of this approach is that the accuracy of the classification result is profoundly dependent on the design of the feature extraction stage. In recent years, deep learning has developed to be a convenient, effective and robust tool to extract features from images, audio, etc, which does not require handcrafted features. Especially, DCNNs for image classification tasks achieved state-of-the-art results in the ImageNet Large Scale Visual Recognition Challenge since 2012 [23].

ResNet. In theory, the performance of the neural network should be positively related to the depth of the network, because the deeper the network, the more parameters it has, the more complicated it is. But the early experimenters observe that as the number of network layers increases, the model accuracy will rise first and then reach saturation, and continuing to increase layers will result in a decrease in accuracy sharply, which we called degradation [17]. ResNet's [17, 18, 32, 35] proposal solves this problem very well. By introducing the residual network structure, it can make the network very deep, at the same time the final classification result is also very satisfactory. In this case, the depth of the network can be extended to tens, hundreds or even thousands of layers, providing the feasibility of extracting and classifying high-level semantic features. ResNet made a stunning appearance in the ILSVRC2015 competition, which raised the

network depth to 152 layers, reducing the error rate to 3.57. In terms of image recognition error rate and network depth, it has greatly been improved compared with previous models. This also makes the network become the backbone network of the later convolutional neural network model, and many models with excellent performance subsequently are transformed on the basis of ResNet [11, 10, 16, 26, 15, 37].

Cambricon. CPU and GPU are designed to handle many different computing tasks originally. They have multiple functional logic units inside, which are widely applicable. But for a computationally intensive computing task, they are not so efficient [1, 4]. Current artificial intelligence algorithm mainly includes two aspects: convolutional neural network and recurrent neural network. From the point of view of decomposition, they are composed of a lot of matrix multiplication or tensor element-by-element multiplication, so the CPU will no longer be suitable for this algorithm, and the GPU will be better, but there is still a lot of room for improvement. Since the introduction of the Cambricon chip, it has sparked a wave of research and application of deep learning accelerators. It is designed for the local and computational characteristics of artificial intelligence algorithms and neural network models to achieve better performance acceleration ratio and computing capability consumption ratio. On this basis, the application scenarios targeted by the deep learning chip are further divided, so the high-performance computing architecture DaDianNao [3] for the server side, the ShiDianNao [6] for the edge-end device application scenario, the PuDianNao [28] for the more generalized machine learning algorithms, all appeared. And the Cambricon instruction set [29] for a wider range of machine learning accelerators and the Cambricon-X [36] for hardware acceleration using data sparsity have been proposed for better use of these architectures.

3 Experiments

3.1 Multiple Operating Modes of Cambricon

To support the Cambricon machine learning processor, Cambricon modifies the open source deep learning programming framework Caffe, and adds some functions like offline, multi-core forward inference and so on, to form Cambricon Caffe. It is compatible with native Caffe's python/C++ programming interface and the native Caffe network model [20]. Besides, it provides a convenient interface to run various types of deep learning applications and a series of APIs provided by the Cambricon Neuware Machine Learning Library (CNML) for efficient inference. CNML interacts with the Cambricon machine learning processor by calling the Cambricon Neuware Runtime Library (CNRT) and drivers. Applications can also call CNML or CNRT directly to use the Cambricon Machine Learning Processor [29].

In the Cambricon operating environment, a variety of different programming models are supported. Firstly, the Cambricon machine learning processor is a multi-core processor architecture with 32 cores and supports two parallel modes:

model parallelism and data parallelism. The setting of two parallelism parameters is based on the specific model. Reasonable model parallelism and data parallelism can optimize the performance of MLU (Machine Learning Unit). The MLU also supports multi-card mode, such as multiple MLU cards installed on a single server, allowing the model to run on different cards or distributing the calculations to multiple MLU cards. MLU supports two different modes of operating mode: online and offline. Online mode refers to the mode that depends on the Cambricon Caffe framework to run, and offline mode refers to the mode that uses the runtime function interface directly from the framework.

3.2 Design Details

In this work, our main concern is the running time of the model, that is, selecting the most suitable model settings, and completing the inference process of the ResNet101 classification model on the CIFAR-10 dataset in the shortest time. As we mentioned above, in order to speed up the inference process of the model, we try to select multi-core, multi-card, offline operation mode.

First of all, for programming model, we can choose multi-core and multi-card. The multi-core is determined by model parallelism and data parallelism. MLU provides us with up to 32 cores. The degree of parallelism of model and data decides the number of cores used. We use grid computing to choose the best combination of model parallelism and data parallelism for this problem. On the other hand, the BenchCouncil provides only one MLU in the runtime environment, we can't try the multi-card programming process. In terms of the model's operating mode, the online process depends on the operation of the Cambricon Caffe framework, while the offline mode is independent of the framework, so we choose the offline mode. The final result of the operation is shown in the figure 1.

3.3 Results in Challenge

In the subsequent experiments, we test model inference on pre-trained ResNet101 with the CIFAR-10 dataset. The three parameters that need to be adjusted are: parallelism of the model, parallelism of the data, and number of threads. The degree of parallelism of model and data together determines the number of cores used by the model. Because BenchCouncil provides us with a total of 32 cores, single card MLU, in order to get the best inference time, we use the most number of cores. In addition, we did some experiments to find out the influence of threads on the speed of model inference. In order to fully use the 32 cores, our model parallel number and data parallel number will be set as 2/16, 4/8, 8/4, 16/2 and 32/1 in order. Meanwhile, we select 2, 4, 6, ..., 14, 16 to test the optimal number of threads. The experimental results are shown in figure 1. According to it, we can summarize as fllow:

With the same degree of parallelism, the more threads, the longer the time of model inference: we believe that when the degree of parallelism of the model and the data both are 1, a total of 32 threads can be created, so the degree of parallelism and threads are mutually exclusive. Now, in order to maximize

the number of cores, the degree of parallelism is also set to a maximum of 32, and then increase the number of threads, which will cause resource preemption among different threads, reducing the speed of reasoning.

Under the specific number of threads, when the model parallelism and data parallelism are 4 and 8, model inference time is the shortest, and it can be considered that the performance matching of the two factors is optimal.

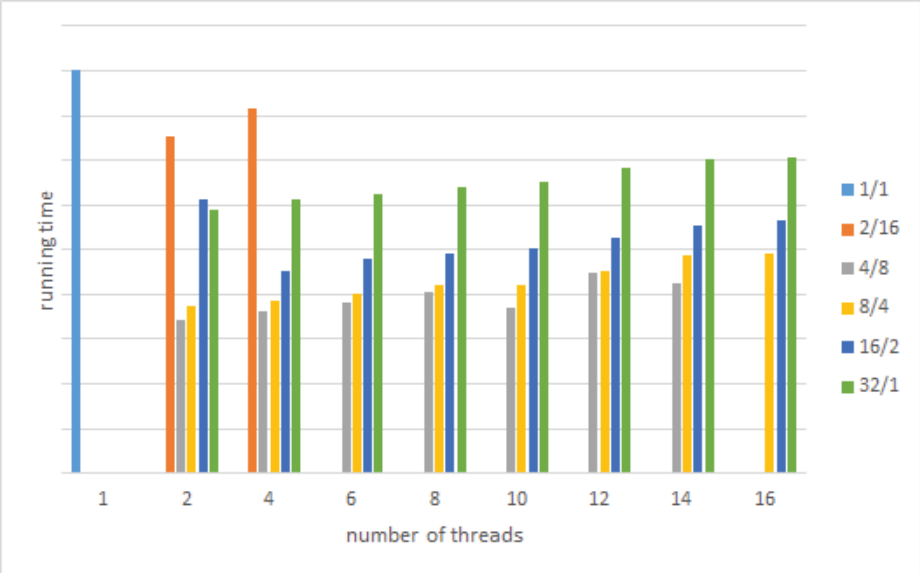

Fig. 1. Illustration of results of model inference under the influence of three factors. The abscissa represents the number of threads, and the ordinate represents the time of model inference. Different colors are different model and data parallelism matching. For example, 2/16 indicates that the model parallelism is 2 and the data parallelism is 16. Specifically, the first one is our baseline. Under certain thread numbers, the combination of different parallelism will cause MLU out of memory. For example, in 16 threads, if 2/16 is used, the memory will overflow, we don't display these results in this figure.

4 Conclusion

In order to solve the problem of image classification on Cambricon, we find out a model setting that is most suitable for the execution of ResNet101 network, and adopt the multicore and multithreading method to transplant the network to the Cambricon operating environment for faster speed. Our transplantation is effective and efficient. We also record the differences among these combinations, and hope the implementation details publicly available can help the community

adopt these useful strategies for object detection, scene parsing and semantic segmentation and advance related techniques.

5 Acknowledgements

The authors would like to thank BenchCouncil for BenchCouncil Testbed. The whole team is supported by Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No.XDA19020400), Equipment Pre-Research Fund (Grant No.61403120405, Grant No.6141B07090131) and Spaceborne Equipment Pre-Research Project(Grant No. 305030704).

References

1. James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: A cpu and gpu math compiler in python. In *Proc. 9th Python in Science Conf*, volume 1, pages 3–10, 2010.
2. Maosen Chen, Tun Chen, and Qianyun Chen. An efficient implementation of the als-wr algorithm on x86 cpus. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
3. Yunji Chen, Tao Luo, Shaoli Liu, Shijin Zhang, Liqiang He, Jia Wang, Ling Li, Tianshi Chen, Zhiwei Xu, Ninghui Sun, et al. Dadiannao: A machine-learning supercomputer. In *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 609–622. IEEE Computer Society, 2014.
4. Adam Coates, Brody Huval, Tao Wang, David Wu, Bryan Catanzaro, and Ng Andrew. Deep learning with cots hpc systems. In *International conference on machine learning*, pages 1337–1345, 2013.
5. Weixin Deng, Pengyu Wang, Jing Wang, Chao Li, and Minyi Guo. Psl: Exploiting parallelism, sparsity and locality to accelerate matrix factorization on x86 platforms. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
6. Zidong Du, Robert Fasthuber, Tianshi Chen, Paolo Ienne, Ling Li, Tao Luo, Xiaobing Feng, Yunji Chen, and Olivier Temam. Shidiannao: Shifting vision processing closer to the sensor. In *ACM SIGARCH Computer Architecture News*, volume 43, pages 92–104. ACM, 2015.
7. Clément Farabet, Berin Martini, Benoit Corda, Polina Akselrod, Eugenio Culurciello, and Yann LeCun. Neuflow: A runtime reconfigurable dataflow processor for vision. In *CVPR Workshops*, pages 109–116, 2011.
8. Wanling Gao, Chunjie Luo, Lei Wang, Xingwang Xiong, Jianan Chen, Tianshu Hao, Zihan Jiang, Fanda Fan, Mengjia Du, Yunyou Huang, Fan Zhang, Xu Wen, Chen Zheng, Xiwen He, Jiahui Dai, Hainan Ye, Zheng Cao, Zhen Jia, Kent Zhan, Haoning Tang, Daoyi Zheng, Biwei Xie, Wei Li, Xiaoyu Wang, and Jianfeng Zhan. Aibench: towards scalable and comprehensive datacenter ai benchmarking. In *2018 BenchCouncil International Symposium on Benchmarking, Measuring and Optimizing (Bench18)*, pages 3–9. Springer, 2018.
9. Wanling Gao, Fei Tang, Lei Wang, Jianfeng Zhan, Chunxin Lan, Chunjie Luo, Yunyou Huang, Chen Zheng, Jiahui Dai, Zheng Cao, Haoning Tang, Kunlin Zhan, Biao Wang, Defei Kong, Tong Wu, Minghe Yu, Chongkang Tan, Huan Li, Xinhui

Tian, Yatao Li, Gang Lu, Junchao Shao, Zhenyu Wang, Xiaoyu Wang, and Hainan Ye. Aibench: An industry standard internet service ai benchmark suite. *arXiv preprint arXiv:1908.08998*, 2019.
10. Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
11. Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.
12. Tongyan Gong and Niu Huiqian. An implementation of resnet on the classification of rgb-d images. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
13. Tianshu Hao, Yunyou Huang, Xu Wen, Wanling Gao, Fan Zhang, Chen Zheng, Lei Wang, Hainan Ye, Kai Hwang, Zujie Ren, and Jianfeng Zhan. Edge aibench: Towards comprehensive end-to-end edge computing benchmarking. *2018 BenchCouncil International Symposium on Benchmarking, Measuring and Optimizing (Bench18)*, 2018.
14. Tianshu Hao and Ziping Zheng. The implementation and optimization of matrix decomposition based collaborative filtering task on x86 platform. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
15. Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
16. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.
17. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
18. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.
19. Pengpeng Hou, Jiageng Yu, Yuxia Miao, Yang Tai, Yanjun Wu, and Chen Zhao. Rvtensor: A light-weight neural network inference framework based on the risc-v architecture. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
20. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.
21. Zihan Jiang, Wanling Gao, Lei Wang, Xingwang Xiong, Yuchen Zhang, Xu Wen, Chunjie Luo, Hainan Ye, Xiaoyi Lu, Yunquan Zhang, Shengzhong Feng, Kenli Li, Weijia Xu, and Jianfeng Zhan. Hpc ai500: A benchmark suite for hpc ai systems. *2018 BenchCouncil International Symposium on Benchmarking, Measuring and Optimizing (Bench18)*, 2018.
22. Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.
23. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

24. Guangli Li, Xueying Wang, Xiu Ma, Lei Liu, and Xiaobing Feng. Xdn: Towards efficient inference of residual neural networks on cambricon chips. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
25. Jiansong Li and Zihan Jiang. Performance analysis of cambricon mlu100. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
26. Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
27. Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
28. Daofu Liu, Tianshi Chen, Shaoli Liu, Jinhong Zhou, Shengyuan Zhou, Olivier Teman, Xiaobing Feng, Xuehai Zhou, and Yunji Chen. Pudiannao: A polyvalent machine learning accelerator. In *ACM SIGARCH Computer Architecture News*, volume 43, pages 369–381. ACM, 2015.
29. Shaoli Liu, Zidong Du, Jinhua Tao, Dong Han, Tao Luo, Yuan Xie, Yunji Chen, and Tianshi Chen. Cambricon: An instruction set architecture for neural networks. In *ACM SIGARCH Computer Architecture News*, volume 44, pages 393–405. IEEE Press, 2016.
30. Chunjie Luo, Fan Zhang, Cheng Huang, Xingwang Xiong, Jianan Chen, Lei Wang, Wanling Gao, Hainan Ye, Tong Wu, Runsong Zhou, and Jianfeng Zhan. Aiot bench: Towards comprehensive benchmarking mobile and embedded device intelligence. *2018 BenchCouncil International Symposium on Benchmarking, Measuring and Optimizing (Bench18)*, 2018.
31. Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
32. Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang. Residual attention network for image classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
33. Yifan Wang, Chen Zeng, and Chundian Li. Exploring the performance bound of cambricon accelerator in end-to- end inference scenario. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
34. Xingwang Xiong, Xu Wen, and Cheng Huang. Improving rgb-d face recognition via transfer learning from a pretrained 2d network. In *International Symposium on Benchmarking, Measuring and Optimization (Bench19)*. Springer, 2019.
35. Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
36. Shijin Zhang, Zidong Du, Lei Zhang, Huiying Lan, Shaoli Liu, Ling Li, Qi Guo, Tianshi Chen, and Yunji Chen. Cambricon-x: An accelerator for sparse neural networks. In *The 49th Annual IEEE/ACM International Symposium on Microarchitecture*, page 20. IEEE Press, 2016.
37. Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

38. Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.