XDN: Towards Efficient Inference of Residual Neural Networks on Cambricon Chips

Guangli Li[1,2*], Xueying Wang[1,2*], Xiu Ma[1,3,4*],
Lei Liu[1(✉)], and Xiaobing Feng[1,2]

[1] State Key Laboratory of Computer Architecture,
Institute of Computing Technology, Chinese Academy of Sciences, China
[2] School of Computer Science and Technology,
University of Chinese Academy of Sciences, China
[3] College of Computer Science and Technology, Jilin University, China
[4] MOE Key Laboratory of Symbolic Computation and Knowledge Engineering,
Jilin University, China
{liguangli, wangxueying, maxiu01, liulei, fxb}ict.ac.cn
These authors contributed equally to this work

Abstract. In this paper, we present XDN, an optimization and inference engine for accelerating residual neural networks on Cambricon chips. We leverage a channel pruning method to compress the weights of ResNet-50. By exploring the optimization opportunities in computational graphs, we propose a layer fusion strategy, which dramatically decreases the number of scalar computation layers, such as Batch Normalization, Scale. Furthermore, we design an efficient implementation of XDN, including data preprocessing, hyper-parameter auto-tuning, etc. The experimental results show that the ResNet-50 model can achieve significant speedup without accuracy loss by using our XDN engine.

Keywords: Artificial Intelligence Systems · Residual Neural Networks · Cambricon Chips · Performance Optimization.

1 Introduction

Recent years, neural networks have been achieved remarkable performance in various areas, including image classification [17, 21, 12], object detection [8, 7, 20], etc. However, the deep neural networks with increasing computational complexity are inefficient on traditional general-purpose hardware such as CPU. Accordingly, research on machine learning specific processors becomes an inevitable trend to satisfy the ever-increasing demand on computation capability in artificial intelligent domains [9]. A batch of domain-specific processors have been developed, such as Cambricon chips [1, 3, 2]. *2019 BenchCouncil International AI System and Algorithm Challenges* include the system challenges on Cambricon [18], X86 [4, 11], RISC-V [13] and a 3D face recognition algorithm challenge [22], which poses challenges to the optimization of artificial intelligence algorithms and systems.

2 Our Approach

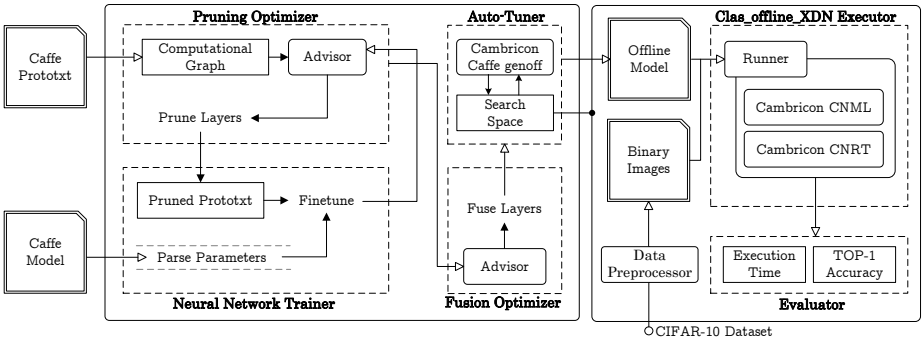

Fig. 1. Overview of XDN Engine

In this paper, we focus on optimizing the performance of ResNet-50 [12], a widely used residual neural network architecture in practice, on Cambricon chips. An optimization and inference engine, namely XDN (XiaoDianNao), is presented, which is composed of a channel pruning approach, fusion strategies, hyper-parameters tuning, data preprocessing, and an efficient implementation executor. Figure 1 demonstrates the approach of our XDN engine.

2.1 Pruning Optimizer and Trainer

The original Caffe prototxt is converted to a computational graph. An Advisor is used to guide the channel pruning and the pruned model is re-trained by the Trainer. The pruning approach by Optimizer and Trainer is iteratively performed until all layers are optimized or unless the accuracy of model is not satisfied.

2.2 Fusion Optimizer

The layers of pruned neural network are fused by an Advisor. The main fusion strategies including: 1) fusing convolution layers with correlative batch normalization layers and scale layers; 2) fusing convolution layers in different branches of building blocks. Then, a fused and pruned neural network model is obtained.

2.3 Auto-Tuner

Cambricon chips supports multi-core parallel execution and have several hyper-parameters, such as the number of threads, model_parallel, and data_paralell. The Auto-Tuner performs all options in the search space and the best option of hyper-parameters, which has minimum execution time, is recorded. The final offline model with the best option is generated by the "genoff" command of the Cambricon Caffe.

2.4 Executor and Evaluator

In order to decrease the time of reading data, the images of CIFAR-10 dataset are preprocessed and converted to a binary file by the Data Preprocessor. We design a Clas_offline_XDN Executor, which executes the final offline model with best hyper-parameters using Cambricon CNML and Cambricon CNRT. Finally, the execution time and the accuracy are calculated by the Evaluator.

3 Experimental Evaluation

3.1 Environmental Setting

We use Cambricon Caffe, a modified BVLC Caffe [14], as the development framework to train, test and generate neural network models. All experiments are performed on a server node of the BenchCouncil New Technology Testbed[1], which is equipped with an Intel i7-4770K CPU and a Cambricon MLU100 accelerator. The Cambricon MLU100 is a multi-core machine learning accelerator which contains 32 cores. The 32 cores are connected to 4 DDR controllers through network-on-chip (NOC) and each DDR controller is connected with 8 cores. In addition, Cambricon also provides the Cambricon Neuware Machine Learning Library (CNML) and the Cambricon Neuware Runtime Library (CNRT) for users to develop deep learning applications.

3.2 Overall Performance

Optimization		Speedup	Accuracy
OPT-0	**Baseline**	1×	0.8439
OPT-1	**OPT-A-1**	4.48×	0.8439
	OPT-B-1	5.04×	0.8462
	OPT-C-1	5.29×	0.8191
OPT-2	**OPT-A-2**	4.67×	0.8441
	OPT-B-2	**7.44×**	**0.8455**
	OPT-C-2	7.62×	0.8203

Table 1. Performance of ResNet-50 with XDN Engine

We evaluate the performance by using the ResNet-50, which is a widely used neural network model, on the AIBench [6, 5, 15, 10, 19] and the dataset is CIFAR-10 [16]. The source code of AIBench is publicly available from http://www.benchcouncil.org/benchhub/AIBench/ (Sign up to get access). Table 1 illustrates the performance of the ResNet-50 model with different optimization options of XDN.

[1] http://www.benchcouncil.org/testbed/index.html

OPT-0 denotes the baseline model and the original executor, which provided in Cambricon Caffe with default hyper-parameters of generation and parallelization.

OPT-1 denotes the optimized model by channel pruning and our XDN executor. OPT-A, OPT-B, OPT-C represent the model without pruning, the model with pruning of 17 layers which without accuracy loss, and the model with pruning of 25 layers which is faster but has slightly accuracy loss, respectively. The results of OPT-1 show that the efficiency of our channel pruning method.

OPT-2 denotes the optimized model by OPT-1 and layer fusion. The denotations of A, B, and C are similar to OPT-1. The results of OPT-2 show that the efficiency of our layer fusion strategy. As can be seen, OPT-B-2 is the best optimization without accuracy loss, which improves the performance by 7.44× over the baseline model with the original executor.

3.3 Analysis and Discussion

The experiment results prove the efficacy of our XDN engine. Firstly, OPT-A-1, which uses the original ResNet-50 model provided by the organizer, illustrates the efficiency of the XDN executor and auto-tuner. For the same model, the XDN executor is faster than original executor by using data preprocessing. The parallel hyper-parameters, such as the number of threads, are selected by an auto-tuning approach in the auto-tuner. Then, the performance improvement of the channel pruning optimizer can be proved by OPT-A, OPT-B, and OPT-C, which with different pruning degrees. As can be seen, the speed increases with more pruned layers. However, the pruned layers also lead to the accuracy loss. OPT-B is the best model without accuracy loss and OPT-C is the faster model which drops about 2% accuracy loss. Finally, OPT-2 confirms the efficacy of layer fusion optimizer. As can be seen, the performance of fused models (OPT-2) are better than the un-fused models (OPT-1).

Optimizations of XDN				**Speedup**
Pruning	**Fusion**	**Data Preprocess**	**Auto-Tuning**	
×	×	×	×	1×
✓	×	×	×	1.75×
✓	✓	×	×	2.72×
✓	✓	✓	×	3.03×
✓	✓	✓	✓	**7.44**×

Table 2. Analysis of OPT-B-2 Performance

We analyze the contributions of each optimization for OPT-B-2 performance, as shown in Table 2. The performance of first row is the baseline, which uses the original model and the default Cambricon Caffe executor, and the performance of last row the OPT-B-2, which is our best optimization method. As can be seen, each optimization method has necessity and effectiveness, which contributes the final performance in the XDN engine.

4 Conclusion

In this paper, we presented an optimization and inference engine, namely XDN, for accelerating deep neural networks on Cambricon chips. The experimental results show that the optimized model can achieve high speedup without accuracy loss. The optimization methods of the XDN can not only be used to optimize residual neural networks, such as ResNet-50, but also other deep neural networks, such as GoogLeNet. In the future work, we plan to formalize our approach and test more deep neural network models.

Acknowledgment

This work is supported by the National Key R&D Program of China under Grant No.2017YFB1003103, the Key Program of National Natural Science Foundation of China under Grant No.61432016, and the Science Fund for Creative Research Groups of the National Natural Science Foundation of China under Grant No.61521092.

References

1. Chen, T., Du, Z., Sun, N., Wang, J., Wu, C., Chen, Y., Temam, O.: Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning. In: ACM Sigplan Notices. pp. 269–284. ACM (2014)
2. Chen, Y., Chen, T., Xu, Z., Sun, N., Temam, O.: Diannao family: energy-efficient hardware accelerators for machine learning. Communications of the ACM pp. 105–112 (2016)
3. Chen, Y., Luo, T., Liu, S., Zhang, S., He, L., Wang, J., Li, L., Chen, T., Xu, Z., Sun, N., et al.: Dadiannao: A machine-learning supercomputer. In: Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture. pp. 609–622. IEEE Computer Society (2014)
4. Deng, W., Wang, P., Wang, J., Li, C., Guo, M.: Psl: Exploiting parallelism, sparsity and locality to accelerate matrix factorization on x86 platforms. In: International Symposium on Benchmarking, Measuring and Optimization (Bench'19). Springer (2019)
5. Gao, W., Luo, C., Wang, L., Xiong, X., Chen, J., Hao, T., Jiang, Z., Fan, F., Du, M., Huang, Y., et al.: Aibench: towards scalable and comprehensive datacenter ai benchmarking. In: International Symposium on Benchmarking, Measuring and Optimization. pp. 3–9. Springer (2018)
6. Gao, W., Tang, F., Wang, L., Zhan, J., Lan, C., Luo, C., Huang, Y., Zheng, C., Dai, J., Cao, Z., et al.: Aibench: an industry standard internet service ai benchmark suite. arXiv preprint arXiv:1908.08998 (2019)
7. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015)
8. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 580–587 (2014)

9. Guo, K., Zeng, S., Yu, J., Wang, Y., Yang, H.: A survey of fpga-based neural network inference accelerators. ACM Transactions on Reconfigurable Technology and Systems (TRETS) **12**(1), 1–26 (2019)
10. Hao, T., Huang, Y., Wen, X., Gao, W., Zhang, F., Zheng, C., Wang, L., Ye, H., Hwang, K., Ren, Z., et al.: Edge aibench: towards comprehensive end-to-end edge computing benchmarking. In: International Symposium on Benchmarking, Measuring and Optimization. pp. 23–30. Springer (2018)
11. Hao, T., Zheng, Z.: The implementation and optimization of matrix decomposition based collaborative filtering task on x86 platform. In: International Symposium on Benchmarking, Measuring and Optimization (Bench'19). Springer (2019)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
13. Hou, P., Yu, J., Miao, Y., Tai, Y., Wu, Y., Zhao, C.: Rvtensor: A light-weight neural network inference framework based on the risc-v architecture. In: International Symposium on Benchmarking, Measuring and Optimization (Bench'19). Springer (2019)
14. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM international conference on Multimedia. pp. 675–678. ACM (2014)
15. Jiang, Z., Gao, W., Wang, L., Xiong, X., Zhang, Y., Wen, X., Luo, C., Ye, H., Lu, X., Zhang, Y., et al.: Hpc ai500: a benchmark suite for hpc ai systems. In: International Symposium on Benchmarking, Measuring and Optimization. pp. 10–22. Springer (2018)
16. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. Tech. rep., Citeseer (2009)
17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
18. Li, J., Jiang, Z.: Performance analysis of cambricon mlu100. In: International Symposium on Benchmarking, Measuring and Optimization (Bench'19). Springer (2019)
19. Luo, C., Zhang, F., Huang, C., Xiong, X., Chen, J., Wang, L., Gao, W., Ye, H., Wu, T., Zhou, R., et al.: Aiot bench: towards comprehensive benchmarking mobile and embedded device intelligence. In: International Symposium on Benchmarking, Measuring and Optimization. pp. 31–35. Springer (2018)
20. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)
21. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
22. Xiong, X., Wen, X., Huang, C.: Improving rgb-d face recognition via transfer learning from a pretrained 2d network. In: International Symposium on Benchmarking, Measuring and Optimization (Bench'19). Springer (2019)