Dylan Markovic

Article Summaries

Parallel Algorithms

C. Yoo and S. Alawneh, "Accelerating 2-D Image Convolution Using a Graphics Processing Unit," 2021 IEEE Western New York Image and Signal Processing Workshop (WNYISPW), Rochester, NY, USA, 2021, pp. 1-5, doi: 10.1109/WNYISPW53194.2021.9661289.

* Image convolutions are expensive in terms of computations, and gpus are usually used to expedite the procedure, the research done is to find possible ways to handle the large amount of data being transferred from cpu to gpu for the process
* Describes the basic process of a matrix convolution
* OpenMP is a library mainly used for parallel cpu computing
* Cuda is a library mainly used for parallel gpu computing
* Experiment uses different threading quantities for cpu and gpu, with different structures of memory(shared/exclusive, etc)
* Memory setup is the limiting factor in terms of gpu convolution calculations

Jin, Peter H. et al. “Spatially Parallel Convolutions.” International Conference on Learning Representations (2018).

* Aims to sidestep the memory limitation of single gpus are sharing the memory
* Tensors are spatially distributed through memory partitions, where each partition is on a single processing unit
* Explains the forward and backward passing using partitioned and non-partitioned methods
* Technique used in paper had excellent scaling in computation time and memory usage

J. Lu, K. Zhang, M. Chen and K. Ma, "Implementation of parallel convolution based on MPI," Proceedings of 2013 3rd International Conference on Computer Science and Network Technology, Dalian, China, 2013, pp. 28-31, doi: 10.1109/ICCSNT.2013.6967057.

* Implements a parallelized convolution computation based on message passing interface.
* Point to point mpi communicates messages directly between two processes
* Collective communication mpi communicates information for multiple processes
* Traditional parallelization of convolutions usually spilts the input/kernel computation between n processors, and then the total value is merged, this needs large data messages and a lot of idle time of processors when the multiplications are not being computed
* This paper uses a matrix partition to try to improve these issues where each partition behaves completely independently until the final merge occurs

Pourghassemi, Behnam & Zhang, Chenghao & Lee, Joo & Chandramowlishwaran, Aparna. (2020). Brief Announcement: On the Limits of Parallelizing Convolutional Neural Networks on GPUs.

* Majority of CNN’s are deployed using pytorch and tensorflow, which accomplish these networks serially
* ~60% of computation time for CNN’s developed using the above Deep Learning frameworks is from the convolution calculations during the training time
* Concurrency of gpu processes have to be assigned to separate executors.
* Results found cuDNN library unable to run multiple convolutions concurrently
* In order to use publicly available libraries to parallelize convolutions, CUDA is not a viable option

Vasudevan, Aravind et al. “Parallel Multi Channel convolution using General Matrix Multiplication.” 2017 IEEE 28th International Conference on Application-specific Systems, Architectures and Processors (ASAP) (2017): 19-24.

* One approach to parallelizing convolutions is called the im2col approach, this method is briefly summarized. The issue with this approach is that the memory requirements get increasingly large as the inputs become larger
* New approach using image data as columns instead of rows
* Treats a convolution of kxk image as k2 convolutions of 1x1 matrices allowing for no data replication
* Their approach had much lower execution time (see bar plots) than direct convolution method, yet similar results for the im2col and im2row techniques

Other Possible documents to review:

1. Tal Ben-Nun and Torsten Hoefler. 2019. Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. ACM Computing Surveys (CSUR) 52, 4 (2019), 1–43.
2. Hongwen Dai, Zhen Lin, Chao Li, Chen Zhao, Fei Wang, Nanning Zheng, and Huiyang Zhou. 2018. Accelerate GPU concurrent kernel execution by mitigating memorypipelinestalls.In2018IEEEInternationalSymposiumonHighPerformance Computer Architecture (HPCA). IEEE, 208–220.
3. XiaZhao,Zhiying Wang,andLievenEeckhout. 2018. Classification-driven search for effective sm partitioning in multitasking GPUs. In Proceedings of the 2018 International Conference on Supercomputing. 65–75.
4. Qiumin Xu, Hyeran Jeon, Keunsoo Kim, Won Woo Ro, and Murali Annavaram. 2016. Warped-slicer: efficient intra-SM slicing through dynamic resource parti tioning for GPU multiprogramming. In 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA). IEEE, 230–242.
5. Zhenning Wang, Jun Yang, Rami Melhem, Bruce Childers, Youtao Zhang, and Minyi Guo. 2016. Simultaneous multikernel GPU: Multi-tasking throughput processors via fine-grained sharing. In 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA). IEEE, 358–369.
6. DanielStrigl,KlausKofler,andStefanPodlipnig.2010. Performanceandscalability of GPU-based convolutional neural networks. In 18th Euromicro International Conf. on Parallel, Distributed and Network-Based Processing. IEEE, 317–324.
7. Linpeng Tang, Yida Wang, Theodore L Willke, and Kai Li. 2018. Scheduling ComputationGraphsofDeepLearningModelsonManycoreCPUs. arXivpreprint arXiv:1807.09667 (2018).
8. Linnan Wang, Jinmian Ye, Yiyang Zhao, Wei Wu, Ang Li, Shuaiwen Leon Song, Zenglin Xu, and Tim Kraska. 2018. Superneurons: Dynamic GPU memory man agement for training deep neural networks. In Proc. of the ACM SIGPLAN Symp. on Principles and Practice of Parallel Programming. ACM, 41–53.
9. G. Lu, W. Zhang, and Z. Wang, "Optimizing GPU memory transactions for convolution operations," 2020 IEEE International Conference on Cluster Computing (CLUSTER), 2020, pp. 399-403, doi: 10.1109/CLUSTER49012.2020.00050.
10. F. N. Iandola, D. Sheffield, M. J. Anderson, P. M. Phothilimthana, and K. Keutzer, "Communication-minimizing 2D convolution in GPU registers,” 2013 IEEE International Conference on Image Processing, 2013, pp. 2116-2120.
11. A. Tousimojarad, W. Vanderbauwhede, and W. P. Cockshott, “2D image convolution using three parallel programming models on the Xeon Phi,” unpublished.
12. S. Goyat and A. Sahoo, “Scheduling algorithm for CPU-GPU based heterogeneous clustered environment using map-reduce data processing,”
13. S. Yu, M. Clement, Q. Snell, and B. Morse, “Parallel algorithms for image convolution,” in Proceedings of the International Conference on Parallel and Distributed Techniques and Applications, Las Vegas, NV, 1998.
14. H. Jin, M. Frumkin, and J. Yan, “The OpenMP implementation of NAS parallel benchmarks and its performance,” NAS Technical Report, Oct. 1999.
15. N. Zhang, Y. Chen, and J. Wang, "Image parallel processing based on GPU," 2010 2nd International Conference on Advanced Computer Control, pp. 367-370, Mar. 2010.
16. D. Hernández, G. Olague, B. Hernández, and E. Clemente, “CUDA-based parallelization of a bio-inspired model for fast object classification,” Neural Computing and Applications, vol. 30, no . 10, pp. 3007–3018, Nov. 2018.
17. Andrew Lavin. 2015. maxDNN: An efficient convolution kernel for deep learning with maxwell GPUs. arXiv preprint arXiv:1501.06633 (2015).
18. Zhihao Jia, Matei Zaharia, and Alex Aiken. 2018. Beyond data and model parallelism for deep neural networks. arXiv preprint arXiv:1807.05358 (2018).
19. Tim Dettmers. How to Parallelize Deep Learning on GPUs. http://timdettmers.com/2014/10/09/deep-learning-data-parallelism/, http://timdettmers.com/2014/11/09/model-parallelism-deep-learning/, 2014.
20. Amir Gholami, Ariful Azad, Peter Jin, Kurt Keutzer, and Aydin Buluc. Integrated Model, Batch and Domain Parallelism in Training Neural Networks. arXiv preprint arXiv:1712.04432, 2018.
21. Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, and Kurt Keutzer. Firecaffe: nearlinear acceleration of deep neural network training on compute clusters. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.
22. Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. arXiv preprint arXiv:1404.5997, 2014
23. Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training Deep Nets with Sublinear Memory Cost. arXiv preprint arXiv:1604.06174, 2016.
24. Jason Jong Kyu Park, Yongjun Park, and Scott Mahlke. 2017. Dynamic resource management for efficient utilization of multitasking GPUs. In Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems. 527–540.
25. M. Afif, Y. Said, and M. Atri, “Efficient 2D convolution filters implementations on graphics processing unit using Nvidia Cuda,” International Journal of Image, Graphics and Signal Processing, vol. 10, no. 8, 2018.