POSTER: Exploring Deep Reuse in Winograd CNN Inference

Ruofan Wu[◇], Feng Zhang[◇], Zhen Zheng[★], Xiaoyong Du[◇], Xipeng Shen[+]
[◇]Key Laboratory of Data Engineering and Knowledge Engineering (MOE), and School of Information, Renmin University of China
[★]Alibaba Group
[+]Computer Science Department, North Carolina State University
2017202106@ruc.edu.cn,fengzhang@ruc.edu.cn,james.zz@alibaba-inc.com,duyong@ruc.edu.cn,xshen5@ncsu.edu

Abstract

Convolutional neural networks (CNNs), as representatives of deep learning, are one of the most commonly used neural networks in applications such as graphic image analysis. However, CNN has heavy computation patterns; network training processes could take several hours even with modern processors. Different from the training process, the inference process is more often executed on devices with low computing power, such as CPUs. Fortunately, a minimal filtering algorithm, Winograd, can reduce the convolution computations by reducing the number of multiplication operations. We find that the Winograd convolution can be further accelerated by reusing the similar data and computation patterns, which is called *deep reuse.*

CCS Concepts: • **Computing methodologies** → **Neural networks**; • **Theory of computation** → *Massively parallel algorithms.*

Keywords: deep reuse, Winograd, CNN, inference

1 Introduction

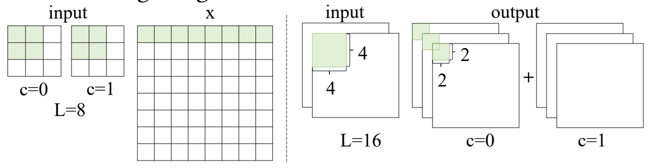
Winograd convolution [3] has been proved to be a very effective optimization for CNNs. By replacing the original convolutions with Winograd minimal filters that have fewer arithmetic operations [6], Winograd convolution greatly reduces the number of computations of CNNs: theoretically, it can reduce the arithmetic complexity by at least 2.25× [3], thus resulting in significant time and energy savings. Most current deep learning libraries support Winograd convolution for CNNs, including Nvidia cuDNN and Intel oneDNN (previously known as MKL-DNN). Furthermore, many efforts have been made to optimize the Winograd convolution by improving hardware efficiency.

Previous studies on Winograd convolution mainly focus on how to optimize the algorithm on specific hardware, such as GPUs, without considering optimizing the structure of the algorithm. Recently, we notice that a novel technique called *deep reuse*, can discover and exploit reusable computations to accelerate convolutions rather than applying traditional code optimizations. By detecting similarities among neuron vectors, this technique reuses intermediate results in CNN inference, and thus saves both space and time on the fly [5]. However, the current deep reuse technique targets only GEMM-based convolutions. If we apply deep reuse to Winograd algorithm, the performance of CNN inference could be further significantly improved.

PPoPP '21, February 27-March 3, 2021, Virtual Event, Republic of Korea
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8294-6/21/02...$15.00
https://doi.org/10.1145/3437801.3441588

2 Similarity in Winograd Convolution

Deep reuse has been proved to be a great success in accelerating the GEMM-based CNN inference process [5]. In this section, we explore the reuse opportunities of deep reuse in Winograd-based convolutions. We show the difference between applying deep reuse in GEMM-based convolution and Winograd-based convolution in Figure 1, and we have the following insights.



(a) *Neuron vector* in GEMM-based deep reuse.

(b) *Tile* in Winograd-based convolution. The input tile size 4x4 and output tile size 2x2 are defined by F(2x2,3x3).

Figure 1. Comparison between GEMM-based deep reuse and Winograd-based deep reuse. x denotes the unfolded input matrix, c denotes the index of the input channel, and L denotes the length of neuron vectors.

First, the neuron similarities in GEMM-based convolution also exist in Winograd convolution. Here, we show the similarity between the *neuron vector* in GEMM-based deep reuse and the *tile* in Winograd convolution. For GEMM-based deep reuse, the neuron vectors are local neurons in the input images or feature maps. Figure 1 (a) illustrates a two-channel input with 2×2 filter in GEMM-based deep reuse and the length of neuron vector is eight. For Winograd-based convolution, as shown in Figure 1 (b), the 4×4 tiles are also local neurons in the input images or feature maps. Since it has been proved that strong similarities exist among neuron vectors in GEMM-based deep reuse [5], there is a high probability that similarities also exist among tiles in Winograd convolution (86.1 to 91.6% similarity in our experiment, detailed in Section 3).

Second, deep reuse cannot be directly applied to Winograd convolution. Applying deep reuse to Winograd convolutions involves more complexities than that to GEMM-based convolutions. For example, for GEMM-based deep reuse, the length of neuron vectors can be flexibly adjusted by users in order to attain a better accuracy and performance (more details are elaborated in [4, 5]). In contrast, the tiles in Winograd-based convolution are fixed, which is to say, we cannot divide the tiles into different vectors for clustering to retain the advantages of the original Winograd algorithm, because results are obtained from tiles. Therefore, we need to perform deep reuse on the granularity of tiles, as shown in Figure 1 (b).

Third, deep reuse is still worth to be applied to Winograd convolution, because it has much higher performance potentials than that in GEMM-based convolution. Winograd convolution reduces the multiplication computations by 2.25× than the direct convolution or GEMM-based convolution [3]. The great performance benefits make it deserve the effort to be further optimized. Moreover, the tile in Winograd convolution is suitable as the object we reuse, as discussed above. Assume that there are large similarities among tiles of batched input, which is proved by the experiments in Section 3, a large amount of computation can be saved by leveraging the similarities, and higher speedups of Winograd convolution can be achieved.

3 Empirical Observations

Opportunities: Experimental observations prove our assumption on tile similarities: there is huge potential of performance improvement for applying *deep reuse* to Winograd convolution.

We conduct experiments to identify the similarities that exist among tiles. We use the trained model of CifarNet [1] on CIFAR10 [2], and run its inference to perform a series of experimental analysis to show the potential benefits of reuse in Winograd convolution. We apply LSH with different numbers of hash functions to the two convolutional layers of *Conv1* and *Conv2* and record the remaining ratio after clustering. Note that a larger number of hash functions means a more fine-grained clustering, and a smaller remaining ratio indicates larger reusable potential.

Single channel. First, we detect the similarities among tiles in each channel with different batch sizes and report the average remaining ratio of each channel. The results are demonstrated in Figure 2, which shows that the two convolutional layers exhibit similar trends. The remaining ratio reaches an average of 0.084 in the first convolutional layer and 0.139 in the second convolutional layer; they both increase along with the increase of hash size and the decrease of batch size.

Multiple channels. Then, we detect the similarities among tiles of multiple channels. Assume that the input to the convolutional layer has C channels. We evenly divide the C channels into x parts so each part has C/x channels. We

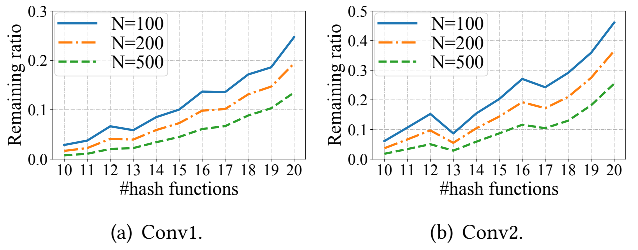


Figure 2. The remaining ratio of the different numbers of hash functions with different batch sizes in a single channel.

treat the tiles of consecutive C/x channels as the neuron vector and apply LSH to them in each part. Figure 3 shows the similarities among tiles of multiple channels with different numbers of channels fixing the batch size as 100. The results show that tiles of multiple channels still generate a small remaining ratio, especially when the number of hash functions is small. Generally, fewer channels lead to a smaller remaining ratio.

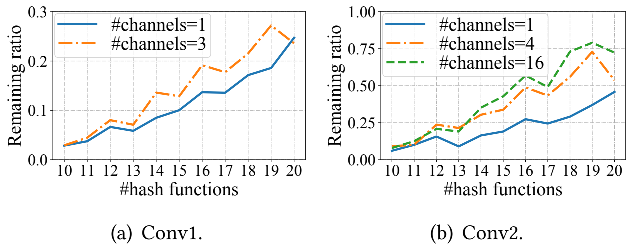


Figure 3. The remaining ratio of the different numbers of hash functions with different numbers of channels.

4 Conclusion

This paper investigates deep reuse with Winograd convolution. By analyzing deep reuse in single channel and multiple channels of Winograd convolution, we find that there is huge potential for applying deep reuse to Winograd convolution.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (No. 2018YFB1004401), National Natural Science Foundation of China (No. 61732014 and 61802412). Feng Zhang is the corresponding author of this paper.

References

[1] 2020. CifarNet. http://places.csail.mit.edu/deepscene/small-projects/TRN-pytorch-pose/model_zoo/models/slim/nets/cifarnet.py.
[2] Alex Krizhevsky. 2012. Learning Multiple Layers of Features from Tiny Images. *University of Toronto* (05 2012).
[3] Andrew Lavin and Scott Gray. 2016. Fast Algorithms for Convolutional Neural Networks. In *CVPR*.
[4] L. Ning, H. Guan, and X. Shen. 2019. Adaptive Deep Reuse: Accelerating CNN Training on the Fly. In *ICDE*.
[5] Lin Ning and Xipeng Shen. 2019. Deep Reuse: Streamline CNN Inference on the Fly via Coarse-Grained Computation Reuse. In *ICS*.
[6] S. Winograd. 1980. Arithmetic complexity of computations.