Title: The Indirect Convolution Algorithm

JX-Ma

2024/9/15

ABSTRACT

when a larger filter tensor will run convolution, we usually need use im2col or im2row transformation to adapt to GEMM algorithm. Indirect convolution does not require data rearrangement but introduces an indirect buffer. The memory overhead and channel dimension used in indirect convolution have a significant impact.

INTRODUCE

On direct convolution, it is difficult to achieve a universal algorithm to optimize each convolution because the input tensor, convolution kernel, and stride of each convolution are different. Therefore, there are two main methods in deep learning libraries to optimize direct convolution. One approach is to optimize convolutions with only a few common parameters, while others default to using regular direct convolution algorithms. Another approach is to automatically generate code based on the input parameters.

The gemm based convolution algorithm first transforms the convolution problem into a GEMM problem and uses a highly optimized BLAS library. There will be a lot of additional memory overhead involved.

Fast Convolution Algorithms use Fourier or Wino- grad transformations to reduce the computational com- plexity of Convolution with large kernel sizes. the algorithmic speedup is limited to specific Convolution parameters.

This paper propose a new convolution algorithm, it eliminates expensive and memory-intensive im2col transformations and allows to replace the im2col buffer with a much smaller indirection buffer.

The algorithm is optimized for NHWC layout and has limited applicability to backward pass of Convolution operator and to the Transposed Convolution operator. The Indirect Convolution algorithm is not efficient for depthwise convolutions.

The Indirect Convolution algorithm

The paper use two modifications that jointly make the GEMM primitive directly suitable for the convolution implementation.

- Removing the assumption that rows of matrix A are separated in memory by a constant stride. Instead, pointers to rows of matrix A are loaded from an array of pointers provided by the caller and denoted indirection buffer. In the context of a 2D convolution, the indirection buffer specifies the address of a row of pixels in the input tensor that contribute to the computation of the output pixel.
- Add an extra loop over elements of the kernel.

Later, the paper introduce the indirect caching ,The Indirection buffer is a buffer of pointers to rows of input pixels. Each row has C pixels, and the rows can optionally be strided.

The Indirection buffer depends on several parameters: shapes of input, output, and filter tensors, convolution stride, dilation, and implicit padding, and pointers to input tensor and explicit zero tensor, and stride of pixel rows in the input tensors.

CONCLUSION

This paper mainly draws three conclusions.

- The Indirect Convolution algorithm is a modification of GEMM-based Convolution algorithms where the GEMM operation reads addresses of rows in the input tensor from indirection buffer.
- Indirect Convolution algorithm offers the universality of GEMM-based algorithm, but with smaller memory footprint and elimination of im2col transformation cost.
- The Indirect Convolution algorithm potentially has interesting performance characteristics beyond the scope of this paper.

SUMMARY

After reading this paper, I know that in order for a convolutional layer to have good performance, we need to optimize it specifically based on some parameters, such as the dimensionality of the filter tensor, especially the width and height, and the convolution step size. For example, in the CUDNN document I previously read, there are also some restrictions on the selection of algorithms in CUDNN. For example, the Winograd algorithm requires a stride of 1, while the FFT algorithm requires the width and height of the filter tensor to be 3x3 or 5x5 The optimization we previously did on the CPU had different block size limitations for each convolutional layer, so there are 12 methods for 12 convolutions. We can consider designing an algorithm that can adaptively select the algorithm.