note of 'Minimizing Computation in Convolutional Neural Networks'

zxp

December 7, 2024

1 content

This is a 2014 paper with less content and is divided into two parts, the first part is a review of CNN and the second part is an optimization of matrix multiplication.

1.1 BACKGROUND

At the time of this paper, CNN are already widely used and the future is promising. The paper states that "CNN is a set of different feature maps. The main working principle of CNN is to gradually extract local features from high-resolution feature maps and then combine these features into more abstract feature maps at lower resolutions.". The paper experimentally calculated the time taken by each part of the CNN and showed that convolutions took the most time. The paper points out that there are two ways to accelerate convolution, using hardware acceleration (GPU or FPGA) or optimizing the calculation. The paper's work is the second one, where the authors found a way to reduce the number of computations.

1.2 CNN

The core of the paper is to use the Strassen algorithm to the convolution (quote 11). Here are a few formulas that can be used to compute the convolution result in fewer calculations: If the formula is minimum, if the length and width of the matrix is a multiple of 2, it can be split until the minimum, if it is not a multiple of 2, it can be filled to a multiple of 2 by padding.

$$W = \begin{pmatrix} W_{1,1} & W_{1,2} \\ W_{2,1} & W_{2,2} \end{pmatrix}, \; X = \begin{pmatrix} X_{1,1} & X_{1,2} \\ X_{2,1} & X_{2,2} \end{pmatrix}, \; Y = \begin{pmatrix} Y_{1,1} & Y_{1,2} \\ Y_{2,1} & Y_{2,2} \end{pmatrix}.$$

Figure 1:

$$\begin{aligned} Y_{1,1} &= W_{1,1} \times X_{1,1} + W_{1,2} \times X_{2,1} \\ Y_{1,2} &= W_{1,1} \times X_{1,2} + W_{1,2} \times X_{2,2} \\ Y_{2,1} &= W_{2,1} \times X_{1,1} + W_{2,2} \times X_{2,1} \\ Y_{2,2} &= W_{2,1} \times X_{1,2} + W_{2,2} \times X_{2,2} \end{aligned}.$$

Figure 2:

$$\begin{aligned} M_1 &:= (W_{1,1} + W_{2,2}) \times (X_{1,1} + X_{2,2}) \\ M_2 &:= (W_{2,1} + W_{2,2}) \times X_{1,1} \\ M_3 &:= W_{1,1} \times (X_{1,2} - X_{2,2}) \\ M_4 &:= W_{2,2} \times (X_{2,1} - X_{1,1}) \\ M_5 &:= (W_{1,1} + W_{1,2}) \times X_{2,2} \\ M_6 &:= (W_{2,1} - W_{1,1}) \times (X_{1,1} + X_{1,2}) \\ M_7 &:= (W_{1,2} - W_{2,2}) \times (X_{2,1} + X_{2,2}) \end{aligned}.$$

Figure 3:

$$\begin{aligned} Y_{1,1} &= M_1 + M_4 - M_5 + M_7 \\ Y_{1,2} &= M_3 + M_5 \\ Y_{2,1} &= M_2 + M_4 \\ Y_{2,2} &= M_1 - M_2 + M_3 + M_6. \end{aligned}$$

Figure 4:

1.3 Experimental

In general, Strassen algorithm has no advantage, and even negative optimization when the matrix is very small, but Strassen algorithm performs better when the matrix size is very large

2 Feelings

The paper also speeds up convolutions by reducing the number of calculations, but only when the matrix is large. A review of CNNS in the first half of the paper can be found in