HLS 를 이용한 Depthwise Separable 컨볼루션 가속기의 FPGA 구현

노수민, 박상수, 정기석(교신저자)
smrho0329@naver.com, po092000@hanyang.ac.kr, kchung@hanyang.ac.kr

FPGA Implementation of Depthwise Separable Convolution Accelerator using HLS

Soo-Min Rho, Sang-Soo Park, Ki-Seok Chung
Hanyang Univ.

요 약

본 논문은 경량화된 컨볼루션 신경망에서 사용되는 Depthwise Separable 컨볼루션 레이어에 적합한 연산 방법을 활용하는 신경망 가속기를 제안한다. 해당 레이어는 필터의 재사용 가능성이 낮은 특성으로 인하여 다른 컨볼루션 레이어에 비해 낮은 가속 효율을 보인다. 본 논문에서는 제시한 최적화 방법을 통해 가속의 효율을 개선할 수 있었으며, High-Level Synthesis 를 통하여 이를 HW 로 구현하였다. 제안하는 HW 는 Intel i9-7900X CPU 보다 Depthwise 컨볼루션 레이어에서 473 배 이상 빠른 추론 성능을 보였다.

Ⅰ. 서 론

Depthwise Separable 컨볼루션 레이어는 Convolution Neural Network (CNN) 아키텍처에서 사용되는 일반적인 컨볼루션 레이어와는 달리, Depthwise 와 Pointwise 컨볼루션이라 불리는 두 개의 레이어를 사용하여 특징(feature)을 추출한다. 이 방법은 연산량과 파라미터의 수를 줄이는데 장점이 있어 경량화된 CNN 모델에서 주로 사용되고 있다 [1]. 이러한 장점에도 불구하고, 특히 Depthwise 컨볼루션 레이어는 Pointwise 컨볼루션보다 추론 시간이 길기 때문에 성능을 개선할 필요가 있다.

CPU, GPU 와 같은 프로세서에서는 컨볼루션 레이어를 행렬 곱셈의 형태로 연산할 수 있도록 텐서 (tensor)들을 변환하는 전처리 과정을 수행하고, 이를 행렬 곱셈의 형태로 연산한다 [2]. 이러한 방법에서는 컨볼루션 필터를 프로세서의 On-Chip 메모리에 저장하고, 이를 재사용하여 성능을 높이는 방법이 사용되고 있다. 하지만 Depthwise 컨볼루션 레이어는 컨볼루션 필터의 재사용 가능성이 낮기 때문에 Off-Chip 메모리의 접근 횟수가 많고, 대역폭이 낮다면 좋은 성능을 얻는 것은 쉽지 않다 [3].

본 논문에서는 이러한 문제를 해결하기 위해, Depthwise 컨볼루션 레이어를 가속함에 있어서 Off-Chip 메모리의 접근 횟수를 줄이는 방법을 제시한다. 또한 HLS (High-Level Synthesis)를 통해 HW 를 구현하여 성능을 향상시켰으며, 제안하는 HW 의 성능을 평가하기 위해 Intel 社의 CPU 와 성능을 비교하였다.

Ⅱ. 본론

2.1 Depthwise Separable 컨볼루션

Depthwise Separable 컨볼루션 레이어는 Depthwise 컨볼루션 레이어와 Pointwise 컨볼루션 레이어라는 2 개의 레이어로 구성된다 [1]. Depthwise 컨볼루션 레이어에서는 각 입력 특징 맵 (input feature map)마다 하나의 2D 필터를 사용해서 2D 컨볼루션 연산을 수행한다. Pointwise 컨볼루션 레이어는 기존의 컨볼루션 레이어와 기본적으로 동일하나, 1×1 필터를 사용해서 연산을 한다.

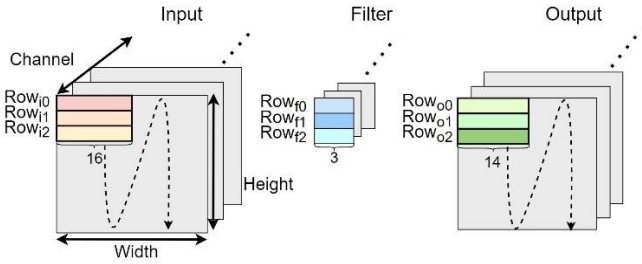
Depthwise 컨볼루션은 각 입력층마다 하나의 필터만을 사용하기 때문에, 기존 컨볼루션 보다 필터의 재사용률이 낮다. 또한 전체적인 연산의 횟수가 적기 때문에, 연산보다는 Off-Chip 메모리 접근 횟수가 성능에 큰 요인으로 작용한다 [3]. 반면, Depthwise 컨볼루션의 입력 특징 (input feature)은 각 입력 층마다 수행되는 연산에서 재사용 될 수 있는 가능성이 존재한다 [3]. 기존 컨볼루션 레이어를 가속하기 위해서 필터를 재사용했다면, Depthwise 컨볼루션 레이어에서는 입력 특징을 재사용하여 성능을 개선하는 것이 가능하다.

2.2 Depthwise 컨볼루션을 위한 특징 재사용 방법

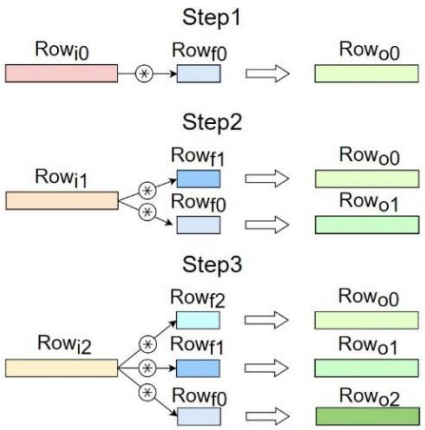
Depthwise 컨볼루션은 중첩되는 영역의 입력 특징 데이터를 재사용하는 것이 가능하다 [3]. 본 논문에서는 보편적으로 이용되는 3×3 크기의 2D 컨볼루션 필터를 사용한 레이어를 대상으로 특징 데이터의 재사용 방법을 적용하였고, 2D 컨볼루션을 각 필터의 행과 입력 벡터의 1D 컨볼루션으로 나누어서 수행하였다.

그림 1(a)는 Depthwise 컨볼루션 연산을 여러 1D 컨볼루션으로 분할한 모습을 보여주며, 그림 1(b)처럼 총 3 단계로 동작한다. 첫 단계에서는 입력 벡터 Row_{i0} 에 접근을 한 뒤, 필터의 Row_{f0} 와 1D 컨볼루션 연산을 수행하여 출력의 부분합 벡터 Row_{o0} 를 생성한다. 다음 단계에서는 인접한 입력 벡터인 Row_{i1} 에 대하여, Row_{f1} 과 1D 컨볼루션 연산을 수행하여 생성한 부분합 벡터를 Row_{o0} 에 누적한다. 이와 동시에 입력 벡터와 Row_{f0} 의 1D 컨볼루션 연산을 수행하여, Row_{o1} 의 부분합 벡터를 생성한다. 마지막으로 Row_{i2} 와 Row_{f2} 와 연산한 결과를 Row_{o0} 에 누적하여, 하나의 완전한 출력 벡터를 생성한다. 이와 동시에 Row_{f2} 를 Row_{f1} 및 Row_{f0} 와 연산하여, Row_{o1} 및 Row_{o2} 의 부분합 벡터를 생성한다. 이러한 1D 컨볼루션과 벡터의 누적 연산 과정을 반복하여, 입력 벡터는 필터의 모든 행과의 연산에 재사용

되고, 출력 벡터를 높이 방향에 따라 순차적으로 생성하는 것이 가능하다.



(a) 1D 컨볼루션을 통한 Depthwise 컨볼루션 연산



(b) 1D 컨볼루션을 통한 특징 재사용

(그림 1) Depthwise 컨볼루션의 특징 재사용 방법

2.3 Depthwise 컨볼루션 가속기의 구현

특징을 재사용한 Depthwise 컨볼루션 연산 방법은 AWS F1 인스턴스 플랫폼에서 Xilinx 社의 Vitis HLS 를 사용하여 구현하였으며, 설계한 HW 는 250MHz 에서 동작한다. 구현된 HW 는 모든 입력층 중 하나의 입력층에 대해서 동작하며, 한 입력층의 모든 연산이 끝나면 다음 입력층을 처리한다. HLS pragma 를 통해, 입력층 내의 단일 입력 벡터를 On-Chip 버퍼로 저장하는 동작과 해당 버퍼를 이용한 1D 컨볼루션 연산을 하는 동작에 하나의 pipeline 을 적용하여 입력의 처리량을 높였다. 그림 1(a)에서 볼 수 있듯이 입력 벡터는 높이 방향으로 우선 접근된다. 높이 방향의 모든 입력 벡터들의 연산을 끝냈을 때, 입력층의 행 내의 다음 입력 벡터에 대해서 동일하게 연산이 진행된다.

1D 컨볼루션은 16 개의 원소를 가지는 입력 벡터 버퍼에서 3 개의 원소로 구성된 벡터 14 개를 추출한 뒤, 3 개의 원소를 가지는 필터의 행 버퍼와 내적 연산을 각각 병렬로 진행하여 부분합을 생성한다. 또한 이를 3 개의 필터 행 버퍼에서 병렬로 수행하여, 한 pipeline interval 내에서 입력 버퍼와 필터의 모든 행 버퍼들의 1D 컨볼루션 연산을 수행하도록 구현하였다.

표 1 은 구현한 HW 의 FPGA 자원 사용도를 보여준다. 본 논문에서 구현한 HW 는 Depthwise 컨볼루션 레이어의 연산량이 적은 특징에 따라 비교적 적은 수의 DSP 만을 사용해서 연산을 처리하였음을 알 수 있다.

(표 1) FPGA 자원 사용도

자원	BRAM	DSP	LUT	FF
사용도	0	61	13546	8248

2.4 실험 결과 및 분석

구현한 HW 의 성능 평가를 위해, Intel i9-7900X CPU 와 구현한 HW 에서의 소요되는 연산 시간을 측정하였다. 실험에서는 표 2 는 실험 대상으로 지정한 MobileNet 의 일부 레이어들의 특징을 보여준다.

(표 2) MobileNet 의 컨볼루션 레이어

	필터 크기	입력 크기
Conv dw1	3 x 3 x 32	112 x 112 x 32
Conv dw5	3 x 3 x 256	28 x 28 x 256
Conv dw13	3 x 3 x 1024	7 x 7 x 1024

표 3 의 결과와 같이, 구현한 HW 와 CPU 에서 Depthwise 컨볼루션의 두 실행 시간은 473 배 이상 차이가 나는 것을 확인하였다. CPU 에서 동작하는 Depthwise 컨볼루션은 텐서를 행렬로 변환하는 전처리 과정과 행렬 곱셈 라이브러리 (OpenBLAS)를 사용하여 구현한 것으로, Depthwise 컨볼루션을 행렬 곱셈으로 계산하는 기존의 방법으로는 충분한 연산 성능을 얻기에 적합하지 않다는 것을 알 수 있다.

(표 3) Depthwise 컨볼루션 레이어 성능 비교

	i9-7900x	Depthwise conv HW	Speed up
Conv dw1	355ms	0.75ms	473x
Conv dw5	272ms	0.4ms	680x
Conv dw13	141ms	0.27ms	522x

Ⅲ. 결론

본 논문에서는 Depthwise Separable 컨볼루션의 특성을 분석하고, 특징 재사용을 활용한 연산 가속화 방법을 제시하였다. 이 방법을 HW 로 구현함으로써, MobileNet 의 컨볼루션 레이어에서 추론 성능이 대폭 향상이 되는 것을 확인하였다. HW 의 구현은 AWS F1 인스턴스 플랫폼 상에서 Vitis HLS 를 사용하였다. 제안하는 HW 는 Intel i9-7900x CPU 대비 473 배 이상 성능 향상이 이루어짐을 확인하였다.

ACKNOWLEDGMENT

이 논문은 2021 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No.2021-0-00131, 제조 검사장비 경량화를 위한 지능형 엣지컴퓨팅 반도체 개발)을 받아 수행된 연구임.

참 고 문 헌

[1] A. Howard et al."MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv preprint arXiv:1704.04861,2017

[2] K. Chellapilla et al. "High Performance Convolutional Neural Networks for Document Processing," Tenth International Workshop on Frontiers in Handwriting Recognition, Université de Rennes 1, Oct 2006, La Baule (France). ffinria-00112631f.

[3] Lu, G, Zhang, W and Wang, Z. "Optimizing Depthwise Separable Convolution Operations on GPUs," IEEE Transactions on Parallel and Distributed Systems. p. 1. ISSN 1045-9219