CNN 네트워크 물리채널 보안 분석을 위한 하드웨어 가속기 설계

이진재* 박종욱** 이준호*** 김호원****

*부산대학교 (대학원생) **부산대학교 (학부생) ***동의대학교 (학부생)

****부산대학교 (교수)

Hardware Accelerator Design for CNN Network Physical Channel Security Analysis

Jin-Jae Lee* Jong-Uk Park** Jun-Ho Lee*** Ho-Won Kim****

*Pusan National University(Graduate student),

**Pusan National University(Undergraduate student)

***Dong-eui University(Undergraduate student)

****Pusan National University(Professor)

요 약

본 연구에서는 CNN 네트워크 물리채널 보안 분석을 위한 하드웨어 가속기를 구현하였다. 실제 딥러닝 가속기에서 사용되는 기법을 적용하여 상용 딥러닝 가속기의 구조와 유사하도록 구현하였고 물리 채널 분석이 용이하도록 구현하여 딥러닝 가속기 보안 분석의 접근성을 높일 수 있을 것으로 기대된다.

주제어 : CNN, 딥러닝 가속기, 물리채널 보안 분석

I. 서론

최근 딥러닝 기술은 의료, 산업, 가전 등의 다양한 분야에서 활용되고 있으며 이러한 딥러닝 기술에 대한 하드웨어 가속화 연구 또한 활발하게 진행되고 있다. 하지만 딥러닝 네트워크의 학습이나 추론 과정에서는 개인의 생체정보나 위치정보와 같은 민감한 데이터를 다루며, 이를 악용하여 생체정보 탈취를 통한 금융서비스 인증 우회, 개인정보 유출을 통한 프라이버시 침해 등의 악의적인 공격이 이루어질 수 있다. 또한, 학습된 딥러닝 네트워크 모델을 물리채널 분석으로 획득하여 도용하는 방식을 통해 지적재산권을 침해하는 사례도 발생할 수 있다.

딥러닝 가속기에 대한 분석 및 공격기법에 대한 연구는 아직 초기 단계이며, 악의적인 공격을 예방하거나, 공격이 이루어졌을 경우에 대한 대응방안은 딥러닝 가속기 분석에 대한 높은 진입장벽으로 인해 활발한 연구가 이루어지지 못하고 있다.

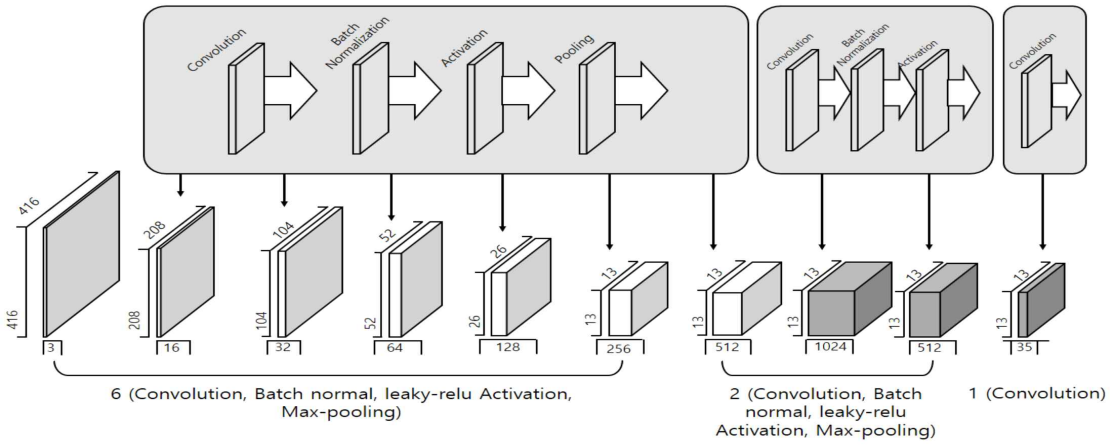
이에 본 연구에서는 물리채널 분석을 위한 딥러닝 가속기를 구현하여 딥러닝 가속기 공격에 대한 대응기술 연구의 접근성을 높이고자 한다.

II. 관련 연구

본 절에서는 물리채널 분석을 통한 CNN 공격과 가속기 설계 관련 연구에 대해 기술한다.

2.1 CNN 설계

가속기 설계에서 가장 중요한 것은 한정된 하드웨어 자원에서 효율적인 연산을 하는 것이다. 또한 가속화를 위해서는 외부 메모리의 접근을 줄이고 on-chip 메모리 사용량을 높여야 한다. 이를 위한 다양한 연산 스케쥴링 방안에 대한 연구가 진행 중이다. 기본적인 개념은 Co



[그림 1] Tiny Yolo 네트워크 구조

nvolution 연산을 할 때, Row base로 partial product들을 이용하여 연산의 파이프라인화, 병렬화 및 메모리 접근을 최소화하는 Row pass[1] 방안 등이 있다.

2.2 물리채널 분석을 통한 CNN 가속기 공격

학습 및 추론 과정에서 사용되는 CNN 가속기는 첫 번째 계층은 많은 이미지를 처리해야 하므로 주로 버퍼를 이용하여 구현이 된다. 이러한 버퍼가 삽입된 첫 레이어 로직의 전력 트레이스를 측정하여 이미지를 재구성하는 방법에 대한 연구가 이뤄지고 있다[2]. 또한 로직의 전력 측정뿐만 아니라 메모리 접근 패턴을 활용하여 각 계층의 이미지 크기, 레이어 수 등 네트워크 구조를 분석함은 물론 가중치 값을 복구 할 수 있는 연구가 활발히 진행 중이다[3]. 이처럼 물리채널 분석을 통한 CNN 가속기 공격은 목표에 따라 다양한 접근법으로 연구되고 있다.

III. 배경지식

3.1 CNN

CNN(Convolutional Neural Network)은 인간의 시신경을 모방하여 만든 딥러닝 구조 중 하나이며 이미지 처리에 높은 성능을 보이는 신경망이다. CNN은 크게 Convolution Layer와 Pooling Layer로 이루어져 있으며 Convolution Layer에서는 이미지의 각 Tensor Value(Height, Width, Channel)를 사용된 kernel에 맞게 합성 곱(convolution) 연산을 통해 축소된 행렬의 결과로 데이터를 도출해내게 된다. Pooling Layer에서는 도출된 합성 곱 데이터의 공간적인 특성을 유지하면서 크기를 줄이는 작업을 실행한다. Pooling에는 Max Pooling, Average Pooling이 있으며 Max Pooling은 kernel과 겹치는 영역 안의 최댓값을 추출하고 Average Pooling은 영역 안의 평균값을 계산해 추출한다.

3.2 Tiny Yolo v2

Tiny Yolo는 기존 Yolo의 알고리즘을 축소하여 저성능 프로세서에서도 원활하게 구동할 수 있게 설계된 알고리즘이다. Yolo는 네트워크의 최종 출력단에서 바운딩 박스의 위치 찾기와 클래스 분류가 동시에 이루어지며 다른 계열(R-CNN)의 딥러닝 네트워크 모델에 비해 간단하고 빠르다는 특징을 가진다.

Tiny Yolo v2 네트워크의 구조는 [그림1]과 같이 $3 \times 416 \times 416$의 입력 데이터를 CNN 기반 알고리즘을 거쳐 $1 \times 125 \times 13 \times 13$의 텐서로 출력시킨다. 여기서 출력된 텐서 내에는 CNN 알고리즘을 통과하면서 계산한 바운딩 박스와 각 바운딩 박스의 정보가 포함된다.

3.3 하드웨어 딥러닝 가속기

하드웨어 딥러닝 가속기(Hardware Deep Learning Accelerator)는 딥러닝 연산의 주를 이루는 병렬 단순 연산(곱셈, 덧셈)을 고속으로 처리하기 위해 사용되는 하드웨어이다. 주로 GPU(Graphic Processing Unit)가 사용되고 있으며 FPGA(Field Programmable Gate Array)와

ASIC(Application-Specific Integrated Circuit)의 형태도 존재한다. ASIC 형태의 하드웨어 딥러닝 가속기에는 NPU(Neural Processing Unit)와 Google의 TPU(Tensor Processing Unit)등이 있다.

IV. CNN 가속기 구조

본 절에서는 Tiny Yolo v2 딥러닝 네트워크 모델의 가속기 설계 구조에 대해서 기술한다.

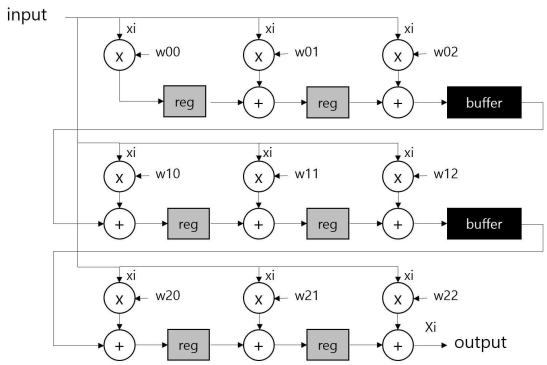
CNN 네트워크의 데이터 양자화와 같은 전처리 연산, 가속기의 연산결과를 사용하여 객체 탐지 결과를 생성하는 후처리 연산은 하드웨어 가속기에서 이루어지지 않으며, 반복이 많고 대용량 연산이 필요한 부분만 하드웨어 가속기상에 구현된다. 본 연구에서 구현하는 CNN 가속기는 16-bit의 fixed point 연산을 지원한다.

4.1 Convolution 모듈

본 연구에서 사용하는 CNN 네트워크 모델인 Tiny Yolo v2는 최대 3x3 크기의 kernel 사이즈를 가지는 filter를 이용하여 convolution 연산을 수행하고, 마지막 fully connected layer에서만 1x1 크기의 kernel 사이즈를 가지는 filter를 사용한다. 때문에 본 연구에서는 filter의 최대 kernel 크기인 3x3을 기본적으로 지원하고 fully connected layer를 위한 kernel 크기의 제어를 컨트롤 유닛에서 수행하는 방식을 가진다. 또한, 대용량 연산이 고속으로 수행되어야 하기 때문에 한 클럭에 하나의 feature map 원소가 출력이 되도록 9개의 MAC(Multiply and Accumulate) 모듈을 사용하는 파이프라인 구조를 가진다.

MAC 모듈은 16-bit fixed point 곱셈과 덧셈을 지원하는 모듈로 파이프라인의 결과 출력 cycle을 단축시키기 위해 단일 clock에 두 연산이 모두 수행되도록 구성된다.

convolution 연산 모듈은 [그림2]와 같은 파이프라인 구조를 가진다. filter의 weight 값은 하나의 채널에 대해서 고정으로 사용되기 때문에 하나의 채널에 대한 연산이 끝날 때까지 register에 저장되며, 매 clock마다 입력 이미지의 pixel 값이 입력되면 해당 자리의 weight 값과의 fixed point 곱셈 연산을 거친 뒤에 이전 값과의 덧셈 연산으로 값을 누적하여 register에 저장한다. 파이프라인이 가득 차게 되면 다음 clock부터 convolution 연산의 결과값이 출력되게 된다.



[그림 2] convolution 연산 모듈 구조

본 연구에서는 연산 코어간의 부채널 신호 간섭을 최소화하여 물리채널 분석이 용이하도록 하기 위해서 convolution 모듈을 하나만 사용한다.

4.2 Batch normalization 모듈

추론용 Batch normalization 연산 모듈은 아래 (a), (b)식과 같은 연산 과정을 가진다.

$$\text{(a)}\ \hat{x_i} = \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}$$

$$\text{(b)}\ y_i = \gamma \hat{x_i} + \beta$$

μ_β(mini-batch mean), σ_β^2(mini-batch variance), γ(scale), β(bias)는 미리 학습된 파라미터 값을 이용하여 연산을 수행하고 (a), (b) 연산을 수행하기 위해서는 fixed point에 대한 square root 연산 모듈이 추가적으로 필요하다. Batch normalization 연산 모듈은 convolution 연산 모듈의 결과값을 입력으로 받아서 연산을 수행하기 때문에 전체 연산 과정이 파이프라인과 같이 동작하기 위해서는 Batch normalization 연산 또한 단일 clock에 연산 결과값을 출력할 수 있어야 한다.

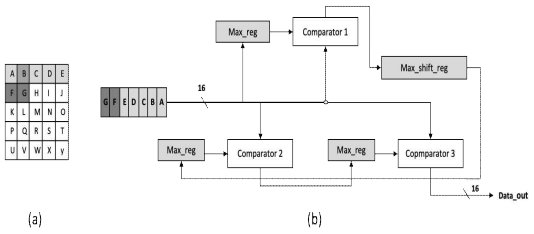
4.3 Activation function 모듈

CNN 관련 Activation function은 하드웨어상에서 MSB의 부호 비트만을 이용하여 효율적인 구현이 가능한 ReLU 및 leaky ReLU function이 주로 사용된다. 기본적인 ReLU의 경우에

는 Multiplexer가 사용되며 leaky ReLU의 경우에는 Scale 값을 곱하는 과정이 있어 DSP(Digital Signal Processor)를 활용하거나 Multiplication 모듈이 추가로 필요하다.

4.4 Pooling 모듈

Max Pooling Layer는 이전 Layer의 출력 데이터의 크기에 따라 유동적으로 설계가 가능하다. 현재 구조는 3개의 Comparator, 3개의 Max Register, (Max image size - kernel size) 크기만큼의 Variable Shift Register를 가진다. 순차적으로 데이터를 입력받기 때문에 kernel의 위치를 기억하고 출력하기 위해 row, column counter가 사용된다. stride의 크기가 3이상이 될 경우 손실되는 데이터가 많기 때문에 주로 1과 2를 사용하는 구조를 가진다.



[그림 3] Max Pooling Layer

(a) 5*5이미지 입력 예시, (b) Max Pooling 구조

[그림3]에서 B 데이터가 3개의 comparator에 입력될 때, Max Register에 들어있는 A와 비교를 한 후 큰 데이터가 Variable Shift register로 이동하여 F 데이터가 comparator에 입력되기 직전에 Max reg에 B의 데이터가 저장된다. 이러한 설계는 순차적인 입력 데이터의 효율적인 처리를 위한 구조이며 외부 메모리의 접근을 최소화한다.

V. 구현

본 연구에서는 CNN 하드웨어 가속기 구현을 위해 Xilinx Artix-7 XC7A200T-1SBG484C FPGA 칩을 사용하는 Digilent Nexys Video board를 사용한다.

CNN 네트워크의 전처리와 후처리 연산은 Intel Core i7-9700K 프로세서를 탑재한 외부의 PC를 사용하여 이루어지고 FPGA와의 연결은 CON-FMC USB인터페이스 보드를 사용하며, BFM(Bus Functional Module)을 사용하여 FPGA 내부의 AXI 버스와 통신이 이루어진다.

CNN 네트워크의 전처리 연산이 수행되면 입력 이미지와 filter값이 FPGA 보드상의 DDR3 메모리에 쓰여지고, 각 레이어의 연산을 위한 파라미터 값은 CNN 가속기 모듈의 CSR(Control and Status Register)에 저장된다. CNN 가속기의 연산 중간값은 고속 연산을 위해 On-chip BRAM(Block RAM)에 저장되고 On-chip BRAM의 크기를 초과하는 레이어 연산 결과의 일부분에 대해서는 외부의 DDR3 메모리에 저장한다.

모든 레이어의 연산이 종료되면 PC에서 CNN 가속기의 연산 결과값을 읽어들여 후처리 연산을 수행한 뒤에 객체 탐지의 결과를 얻을 수 있다.

구현에 있어서 CNN 하드웨어 가속기의 convolution 연산 모듈의 MAC와 같이 고속화가 가능한 부분은 FPGA의 DSP를 사용하여 구현한다.

VI. 결론

본 논문에서는 딥러닝 가속기의 물리채널 보안 분석을 위한 CNN 추론용 하드웨어 가속기를 설계 및 구현하였다. 실제 딥러닝 하드웨어 가속기에서 사용되는 파이프라인 구조를 사용하여 연산 모듈을 구현하였으며, 단일 코어로 구현하여 코어 간의 신호 간섭을 배제하여 물리채널 분석이 용이하도록 구현하였다. 기존의 딥러닝 가속기와 비교하여 분석 난이도를 낮추어 물리채널 보안 분석에 대한 접근성을 높일 수 있을 것으로 기대된다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터지원사업의 연구결과로 수행되었음 (IITP-2020-0-01797)

[참고문헌]

[1] Ding, Caiwen, et al. "REQ-YOLO: A resource-aware, efficient quantization

framework for object detection on FPGAs." Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. 2019.

[2] Lu, Liqiang, et al. "Evaluating fast algorithms for convolutional neural networks on FPGAs." 2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2017.

[3] Yin, Qiao, et al. "FPGA-based High-performance CNN Accelerator Architecture with High DSP Utilization and Efficient Scheduling Mode." 2020 International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS). IEEE, 2020.

[4] Wei, Lingxiao, et al. "I know what you see: Power side-channel attack on convolutional neural network accelerators." Proceedings of the 34th Annual Computer Security Applications Conference. 2018.

[5] Weizhe Hua, Zhiru Zhang, and G. Edward Suh. 2018. Reverse engineering convolutional neural networks through side-channel information leaks. In Proceedingsof the 55th Annual Design Automation Conference, DAC 2018, San Francisco, CA,USA, June, 2018.