Transformer의 Feed-Forward Network 가속화 연구

이창근, 신새빈, 임완수*

성균관대학교 전자전기컴퓨터공학과

*wansu.lim@skku.edu

A Study on Acceleration of the Transformer Feed-Forward Network

Changgeun Lee, Saebin Shin, Wansu Lim

Sungkyunkwan University

요 약

Transformer는 self-attention 기반의 인코더 - 디코더 구조를 사용하여 높은 성능을 보이나, 많은 가중치로 인해 모델 크기가 크고 메모리 사용량이 높아 메모리 자원이 제한된 엣지 디바이스에서 동작이 제약된다. 특히 Feed-Forward Network 레이어는 대규모 행렬 곱 연산과 높은 가중치 비중으로 인해 연산 및 메모리 병목의 주요 원인이 된다. 본 논문은 4비트 대칭 양자화와 니블 패킹 기법을 적용하여 FFN 가중치를 압축하고, 이를 리소스가 제한된 Kria KV260 FPGA의 온칩 메모리에 모두 맵핑하는 방법을 제안한다. Vitis HLS 환경에서 합성한 결과, BRAM 사용량이 3,813개에서 242개로 감소(93.7% 절감)하여 BRAM 한도 내 온칩 맵핑을 달성하였다. 이를 통해 외부 DRAM 접근으로 인한 메모리 병목을 효과적으로 완화하였다.

Ⅰ. 서 론

Transformer는 self-attention기법을 사용하는 인코더와 디코더로 구성된 기계 번역 모델이다[1]. Transformer는 높은 성능을 지니나, 가중치 수가 많아져 모델 크기가 크다. 이는 모델 추론 속도를 저하시킬 뿐만 아니라 메모리 자원이 제한된 엣지 보드에서의 동작을 제약한다. 특히 단어 토큰 임베딩 차원이 큰 경우, 대규모 행렬 곱 연산을 수행하는 Feed-Forward Network(FFN) 레이어가 주요 병목으로 작용한다. 또한 FFN의 가중치는 Transformer 전체 가중치에서 큰 비중을 차지해 메모리 사용량을 크게 증가시킨다. FFN의 연산 병목은 FPGA를 통해 가속할 수 있으나[2], 리소스가 제한된 FPGA 보드에서는 모든 FFN 가중치를 온칩(BRAM/URAM)에 맵핑 시킬 수 없어 메모리 접근 병목(memory bottleneck)이 발생한다.

본 논문에서는 FFN 가중치에 양자화 기법과 니블 패킹을 적용하여 리소스가 제한된 Kria KV260 FPGA의 온칩 메모리에 모든 FFN 가중치를 맵핑 시킴으로써, 메모리 접근 병목을 최소화하는 Transformer 가속화를 제안한다.

Ⅱ. 제안한 가속화 알고리즘

그림1은 FPGA를 통한 FFN 가속화에 대한 전체적인 시스템을 나타낸다. 본 논문에서는 학습이 완료된 FFN 레이어의 모든 가중치들을 FPGA의 온칩 메모리에 효율적으로 맵핑하기 위해, 4비트 대칭 양자화와 니블 패킹을 적용한다. 가중치는 고정된 스케일과 오프셋으로 압축되며 니블 단위로 연속 저장된다. 또한 FFN의 연산 경로는 그대로 유지시켜 정확도 저하를 억제한다.

1. Feed-Forward Network

FFN은 Transformer의 인코더와 디코더 모두에 위치하며 self-attention과 residual connection을 거친 출력에 적용된다. 입력 차원은 $X_{FFN} \in \mathbb{R}^{S \times d_{model}}$이며, 여기서 S는 입력 문장 길이를 의미한다.

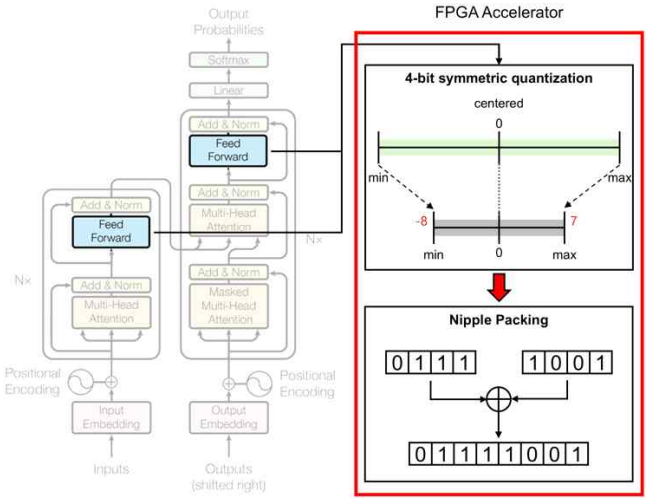

그림 1. FFN 레이어 및 FPGA 가속화를 위한 가중치 압축

FFN 레이어는 2개의 선형 변환과 $W_{hidden} \in \mathbb{R}^{d_{model} \times hidden}$으로 구성된다. 이때 필요한 연산량은 수식(1)과 같다.

$$S \times d_{model}^2 \times hidden + d_{model} \times hidden^2 \times d_{model} = d_{model}^2 \times hidden(S + hidden) \quad (1)$$

FFN은 연산량과 가중치 메모리 지배성을 이유로 병목으로 작용한다. 한 블록의 FFN은 토큰마다 두 차례의 대형 행렬 곱을 수행한다. 특히 어텐션의 주요 연산량은 문장 길이에 비례하기 때문에 엣지 추론 상황에서와 같이 문장 길이를 짧게 유지할 때는 FFN의 비중이 급격히 커져 실제 지연의 주된 원인이 된다. 또한 FFN은 어텐션의 선형 투영을 가중치를 모두 합친 크기와 비슷하거나 더 큰 비중을 차지한다. 이처럼 큰 가중치를 반복적으로 DRAM에서 스트리밍 하는 것은 대역폭과 온칩 메모리(BRAM/URAM) 용량에 영향을 받는다. BRAM 리소스가 낮은 보드에서는 외부 메모리 왕복이 전체 처리량을 제한하는 메모리 병목으로 이어진

표 1. HLS 합성 결과

	BRAM	DSP	FF	LUT
양자화 전	3813	5	18731	5102
양자화 후	242	5	14006	4975

다. 따라서 FFN의 모든 가중치를 온칩에 맵핑시켜 외부 접근을 최소화하여 지연과 처리량을 동시에 개선하고자 한다.

2. 4비트 대칭 양자화

양자화는 부동소수점을 정수 자료형으로 변환하여 모델 크기를 축소하고 추론 속도를 향상하는 기법이다[3,4]. 본 논문에서는 학습 재훈련이 필요 없는 사후 양자화(post-training) 방법을 적용한다. 또한 대칭 스케일(symmetric scale) 방식을 적용하여 가중치를 정수 형태로 변환한다. 대칭 스케일 변환은 수식(2)와 같다.

$$q = clip(round(s \times W), -8, 7), \qquad s = \frac{7}{\max|W|} \tag{2}$$

이후, 가중치 저장 효율을 위해 오프셋을 적용하여 수식(3)의 형태로 변환한다.

$$\tilde{q} = q + 8 \; IN[0, 15] \tag{3}$$

3. 니블 패킹(Nibble Packing)

니블 패킹은 1바이트를 상위 4비트와 하위 4비트로 나누어 각각 별도의 데이터를 저장하는 방식이다. 본 연구에서는 오프셋이 적용된 가중치를 1바이트의 상·하위 니블에 저장하여 메모리 사용량을 절감한다. 이후 부동소수점 복원은 수식(4)와 같다. 본 연구에서는 데이터 패킹과 부동소수점으로 복원하는 과정은 파이프라인으로 겹쳐 실행하여, 언패킹 단계가 지연의 새로운 원인이 되지 않도록 설계한다.

$$W \approx \frac{\tilde{q} - 8}{s} \tag{4}$$

Ⅲ. FPGA 구현

1. 시뮬레이션 세팅

본 시뮬레이션은 FPGA 보드 리소스 내에서 인코더와 디코더의 두 개 FFN 레이어의 모든 가중치를 온칩에 맵핑하는 것을 목표로 하였다. 사용된 FFN 가중치 차원은 512×2048이며, 실험에는 BRAM 자원이 288개로 비교적 제한적인 리소스를 가진 Kria KV260 FPGA 보드를 사용하였다. Transformer는 Pytorch로 학습하였고, FPGA 가속 시뮬레이션은 Vitis HLS 환경에서 수행하였다. Python으로 작성한 PS(Processing System)에서 FPGA로 FFN 레이어의 입력 데이터를 전송하고, FPGA의 PL(Programmable Logic) 측에서 FFN의 행렬곱 연산을 수행하였다. 이후 연산 결과는 다시 PS(Processing System) 측으로 반환되어 트랜스포머 연산의 다음 입력으로 사용되었다.

2. 합성 결과

FFN 가중치는 HLS 커널에서 니블 패킹된 int8 배열로 저장하여 용량을 최대 4배까지 절감하였다. 커널은 니블 데이터에서 상·하위 4비트를 시프트 및 마스킹 연산으로 추출하고, 사전에 정의된 스케일 값을 적용해 복원한 뒤 FFN 연산을 수행하였다. 이때 출력 값은 정확도 유지를 위해 부동소수점으로 유지하였다.

표1은 HLS에서 FFN 레이어 합성 결과를 나타낸다. 양자화 전 BRAM 사용량은 3813개로, KV260의 최대 한도(288개)를 초과하였다. 4비트 양자화 및 니블 패킹을 적용 후 BRAM 사용량이 242개로 줄어, 양자화 전 대비 93.7% 감소하였다. 이를 통해 모든 FFN 가중치를 BRAM 한도 내에 온칩에 맵핑할 수 있었다. 결과적으로 외부 DRAM 왕복으로 인한 메모리 접근 병목을 효과적으로 최소화하였다.

한편 니블 언패킹과 스케일 복원 연산이 조합 논리를 사용함에 따라 LUT 사용량은 증가하고 DSP는 변화가 없었다. 이는 스케일 값을 고정소수점으로 변환해 곱셈 연산을 DSP에 할당하거나, 부분합을 BRAM/URAM으로 이동시키는 방식으로 LUT-DSP-BRAM 간 리소스 균형을 조정할 수 있음을 보였다.

Ⅳ. 결론

본 논문에서는 Transformer의 주요 연산 병목인 Feed-Forward Network(FFN) 가속화를 위해 4비트 대칭 양자화와 니블 패킹 기법을 적용하여, 리소스가 제한된 Kria KV260 FPGA의 온칩 메모리에 FFN의 모든 가중치를 맵핑하는 방법을 제안하였다. Vitis HLS 합성 결과, 제안 기법은 BRAM 사용량을 3,813개에서 242개로 감소시켜(93.7% 절감) BRAM 리소스 내 온칩 맵핑을 가능하게 하였다. 이를 통해 외부 DRAM 접근에 따른 메모리 병목을 효과적으로 완화하였다. 또한 제안된 기법은 모델과 데이터 의존성이 낮아 다양한 Transformer 변형의 FFN에도 재학습 없이 적용할 수 있다. 향후 실제 FPGA 하드웨어 구현 및 Self-Attention 모듈까지 최적화 대상을 확장하여 Transformer 전체의 가속 성능을 극대화할 예정이며, 실제 보드 기준 엔드-투-엔드(end-to-end) 지연, 전력, 온도 프로파일을 계측하여 엣지 상황에서의 이점을 검증할 계획이다.

ACKNOWLEDGMENT

본 연구는 2023년도 산업통상자원부의 재원으로 한국에너지기술평가원(KETEP)과 2024년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받아 수행된 연구임. (RS-2023-00266248, RS-2024-00349885)

참 고 문 헌

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., and Polosukhin, I., "Attention is all you need," Advances in Neural Information Processing Systems 30, 2017.

[2] Li, B., Pandey, S., Fang, H., Lyv, Y., Li, J., and Ding, C., "FTRANS: energy-efficient acceleration of Transformers using FPGA," Proc. ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), 2020.

[3] Choukroun, Y., Kravchik, E., Yang, F., and Kisilev, P., "Low-bit quantization of neural networks for efficient inference," IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2019.

[4] Wu, X., Li, C., Yazdani Aminabadi, R. Y., Yao, Z., and He, Y., "Understanding INT4 quantization for language models: latency speedup, composability, and failure cases," International Conference on Machine Learning (ICML), PMLR, 2023.