거대 언어 모델의 효율적인 추론을 위한 소프트맥스 함수 최적화 방법에 관한 연구

김정현, 정기석*

한양대학교

hanagod2015@hanyang.ac.kr, *kchung@hanyang.ac.kr

A Study on the Optimization of the Softmax Function for Efficient Inference in Large Language Models

Kim, Jeong Hyun, Chung, Ki-Seok*

Hanyang Univ., Seoul, Korea

요 약

트랜스포머 기반의 모델은 기존 모델 대비 우수한 성능으로 많은 인공지능 분야에서 채택되고 있다. 거대 언어 모델 역시 트랜스포머를 기반으로 구성되는 것이 일반적이며 이 같은 모델 구조는 성능에 결정적인 영향을 끼치고 있다. 그러나 트랜스포머의 압도적 성능은 기존 모델 대비 더욱 많은 연산량과 컴퓨팅 성능을 요구한다. 이런 배경에서 트랜스포머를 이루는 다양한 연산 및 계층을 최적화하는 연구들이 진행되고 있다. 본 논문에서는 어텐션 계층에 속한 소프트맥스 함수의 최적화 방법을 제안한다. 그리고 이를 하드웨어로 구현하여 기존의 두 가지 소프트맥스 구현 방법과 비교한다. 결론적으로 본 논문에서 제안하는 최적화 방법은 기존 방법 대비 정확도 대 전력의 비율 측면에서 평균적으로 약 2.31배 더 효율적임을 보였다.

Ⅰ. 서론

기존의 인공지능 모델과 비교했을 때 트랜스포머는 월등히 높은 성능으로 다양한 인공지능 분야에서 채택되고 있는 모델이다. 특히 자연어 처리, 컴퓨터 비전, 추천 시스템과 같은 분야가 대표적인 적용 분야로 꼽힌다. 트랜스포머 모델의 성능 향상의 일례로 컴퓨터 비전의 ImageNet 분류가 있다. 해당 실험에서 트랜스포머 기반의 ViT-G/14 모델과 기존 CNN 기반 모델인 BiT(ResNet-154x4)의 정확도를 비교한 결과 ViT-G/14에서 2.91%의 정확도 향상이 있었다[1]. 트랜스포머 성능 향상의 핵심은 어텐션 메커니즘[2]이다. 하지만 어텐션 메커니즘은 기존 모델보다 더 많은 연산을 요구하고 에너지를 소모한다는 단점이 존재한다[1]. 그렇기에 어텐션 메커니즘의 수학적인 최적화 [3] 혹은 트랜스포머를 구성하는 계층 전반을 최적화하여 트랜스포머 모델의 효율성을 높이는 연구가 활발하게 진행되고 있다. 이런 배경에서 본 논문에서는 어텐션 메커니즘에서 입력값을 확률로 치환하는 역할을 하는 소프트맥스 함수를 최적화한다. 그 이유는 모델의 실행 시간 관점에서 입력 데이터로 사용되는 시퀀스 길이가 증가함에 따라 소프트맥스 함수의 오버헤드가 급격히 커지기 때문이다[4]. 그리고 기존의 소프트맥스 구현 방식과 제안하는 방식을 하드웨어로 구현하여 기존 방식들과 정확도, 전력 그리고 면적을 비교한다.

Ⅱ. 본론

2.1. 어텐션 메커니즘

어텐션 메커니즘은 트랜스포머 기반 모델의 성능 향상에 큰 역할을 하고 있다. 그 이유는 어텐션 메커니즘이 기존 모델의 정보 압축으로 말미암은 정보 손실 문제와 기울기 폭발 및 소실 문제를 해결했기 때문이다[2]. 어텐션 메커니즘은 매 시점에 모든 입력값을 참고한다. 그리고 특정 시점에서 예측해야 할 출력과 연관성 높은 입력값에 집중하여 출력을 예측한다. 일반적으로 입력 값을 '쿼리(*Query*), 전체 입력 값을 '키(*Key*)' 그리고 쿼리와 키의 유사도를 '값(*Value*)'이라고 명명한다. 어텐션 메커니즘은 알고리즘 1.과 같다.

Algorithm 1: Attention Mechanism

Input: $T(Input\ Token)$
Input: $W_Q, W_K, W_V(Query,\ Key\ and\ Value\ Weights)$
Output: $Context$
Parameter: $h(Number\ of\ head), D(Token\ dimension)$
for $0 \leq i \leq h\text{-}1$ **do**
  $Q[i] \leftarrow T[i] \cdot W_Q[i]$
  $K[i] \leftarrow T[i] \cdot W_K[i]$
  $V[i] \leftarrow T[i] \cdot W_V[i]$
  $Attention\ score \leftarrow Q[i] \cdot K[i]^T$
  $Attention\ dist \leftarrow Softmax\left(\frac{Attention\ score}{\sqrt{D}}\right)$
  $Attention\ value \leftarrow Attention\ distribution \cdot V[i]$
end
$Context \leftarrow FFN\left(Concat\left(Attention\ value[i]\right)\right)$

2.2. 교차 소프트맥스 함수

알고리즘 1.의 10번째 줄에서 소프트맥스 함수가 어텐션 분포를 만든다는 걸 알 수 있다. 이러한 소프트맥스 함수는 자연 상수 e의 특정 입력 제곱과 자연 상수 e의 전체 입력 제곱을 모두 더한 값을 나누어 확률을 구한다. 하지만 무리수인 e의 제곱을 하드웨어로 구현하는 건 도전적이다. 기존 방식으로 기억장치를 이용하는 LUT(Look Up Table) 방식과 삼각함수나 로그 같은 연산을 가능하게 하는 CORDIC(COordinate Rotation DIgital Computer)[5] 방식이 존재하지만, 각각 큰 전력 소모와 성능 저하라는 단점이 존재한다. 또한, 거대 언어 모델에서 소프트맥스 함수의 정확도보다 기존 모델 대비 연산량 오버헤드로 인한 전력 소모 증가가 더 주요한 문제로 지적되고 있다[6]. 이에 본 논문에서는 거대 언어 모델 추론을 위한 교차 소프트맥스를 제안한다. 교차 소프트맥스는 임곗값 기반 원-핫 인코딩 방식과 지수 함수의 성질을 이용하는 근사 방식 중 한 개의 결괏값을 출력으로 사용한다. 두 추정 방식 중 임곗값 기반 원-핫 인코딩 방식은 모든 입력값의 가장 큰 값과 다음으로 큰 값의 차이가 임곗값보다 크면

가장 큰 값의 확률을 1로 설정하고 나머지 값들의 확률은 0으로 패딩 하는 방식이다. 임곗값은 실험적으로 자연수 3이 적합하다는 것을 찾아냈다. 다음으로 지수 함수 성질을 이용한 근사 방식은 밑인 자연 상수 e를 자연수 2와 변환하는 방식이다. 그 결과 밑은 자연수 2로 지수는 $log_2 e$로 변환된다. 마지막으로 지수와 입력값의 곱을 반올림한다. 결과적으로 교차 소프트맥스 함수는 기존 방식 대비 하드웨어 친화적인 성질을 갖는다. 앞선 두 가지 방식을 결합한 교차 소프트맥스는 알고리즘 2.와 같은 흐름으로 진행된다.

Algorithm 2: Cross Road Softmax

Input: x
Output: $dist(Distribution)$
Parameter: $k(Number\ of\ input), th(Threshold\ value)$
$1^{st}, 2^{nd} \leftarrow Detect(x)$
if $1^{st} - 2^{nd} \geq 3$ **then**
  $dist[index(1^{st})] \leftarrow 1.0$
  $dist[index(else)] \leftarrow 0.0$
else
  $dist \leftarrow \left(\frac{2^{round(x_i log_2 e)}}{\sum_{k=0}^{i-1} 2^{round(x_i log_2 e)}} \right)$
end

2.3. 교차 소프트맥스 함수의 하드웨어 구현

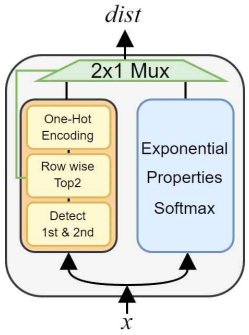

그림 1. 간략화한 교차 소프트맥스 모듈

그림 1.은 교차 소프트맥스 모듈을 간략하게 그린 그림이다. 교차 소프트맥스 모듈은 2.2.장에서 언급한 임곗값 기반 추정 모듈과 지수 함수 성질을 이용한 추정 모듈로 구성되어 있다. 임곗값 기반 추정 모듈인 왼쪽 모듈은 크게 입력 데이터 중에서 가장 큰 값과 두 번째 큰 값은 찾는 모듈, 두 값의 차이를 구하는 모듈, 마지막으로 원-핫 인코딩을 진행하는 모듈로 구성되어 있다. 해당 모듈은 알고리즘 2.의 5번 줄부터 7번 줄까지의 연산을 수행한다. 지수 함수 성질을 이용한 추정 모듈은 그림 1.의 오른쪽 모듈로 알고리즘 2.의 9번 줄에 해당하는 연산을 수행한다. 두 가지 추정 모듈은 병렬적으로 처리되며 최종 결과인 어텐션 분포는 두 값의 차이를 구하는 모듈로부터 선택 신호를 받는 2X1 멀티플렉서를 통해 선택된다.

2.4. 실험 환경 및 실험 결과

실험 결과의 비교를 위해 기존 소프트맥스 함수 구현 방식인 LUT 방식과 CORDIC 방식을 Verilog HDL을 이용하여 구현하였다. 이후 소프트웨어 프레임워크인 PyTorch를 이용하여 거대 언어 모델 벤치마크로 사용되는 GLUE[7]의 CoLA Task로부터 128개의 입력으로 구성된 2,048개의 벡터를 검증 데이터로 추출하였다. 그리고 Vivado Xsim을 이용하여 Verilog HDL로 구현한 세 모듈의 기능 검증을 수행하였다. 이렇게 얻은 세 모듈의 결괏값과 검증 데이터 간의 정확도 비교를 위해 평균 절대 오차(MAE)를 구했다. 또한, Synopsys의 Design Compiler를 이용해 합성을 진행하였다. 합성을 위해 Nangate 45nm OpenCell 라이브러리를 사용했으며 세 가지 모듈 모두 클럭 주파수를 100MHz로 설정하였다. 결론적으로 본 논문에서 제안하는 교차 소프트맥스는 정확도 대 전력 비율(APR)을 비교해 보았을 때 LUT 대비 약 1.57배 CORDIC 대비 약 4.05배 정도의 효율 향상이 있었다.

Table 1: EVALUATION COMPARISON ON SOFTMAX MODULES

Module	Frequency (MHz)	MAE (Acc.)	Power (mW)	Area (um^2)	APR (Acc. Power Ratio)
LUT	100	2.03e-03	4.69	0.2	1
CORDIC	100	1.00e-03	25.44	0.127	0.39
CROSS ROAD	100	1.14e-03	5.30	0.724	1.57

Ⅲ. 결론

본 논문에서는 거대 언어 모델에서 최근 병목현상이 관측되는 소프트맥스 함수를 최적화하는 방법으로 교차 소프트맥스를 제안하였다. 제안하는 교차 소프트맥스는 정확도 대 전력 비율 측면에서 기존 방식인 LUT 방식 대비 약 1.57배의 효율성 향상이 있었으며 CDRDIC 방식 대비 약 4.05배의 효율성 향상 있음을 확인하였다.

ACKNOWLEDGMENT

본 연구는 IDEC에서 EDA Tool를 지원받아 수행하였습니다.

참 고 문 헌

[1] Xiaohua Zhai. Et. Al., “Scaling Vision Transformer.", Conference on Computer Vision and Pattern Recognition, CVPR 2022.

[2] Ashish Vaswanmi. Et. Al., “Attention is All you Need.", Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, NeurIPS 2017.

[3] Mohamed S. Abdelfattah, Et. Al., “Adaptable Butterfly Accelerator for Attention-based NNs via Hardware and Algorithm Co-design”, The International Symposium on Computer Architecture, ISCA 2022.

[4] Jacob R. Stevens. Et. Al., “Softermax: Hardware/Software Co-Design of an Efficient Softmax for Transformers.”, IEEE ACM /IEEE Design Automation Conference, DAC 2021.

[5] Volder, Jack E., “The CORDIC trigonometric computing technique,”, IRE Transactions on electronic computers, vol. 3, pp. 330-334, 1959.

[6] Yubin Qin. Et. Al., “FACT: FFN-Attention Co-optimized Transformer Architecture with Eager Correlation Prediction”, The International Symposium on Computer Architecture, ISCA 2023.

[7] Alex Wang. Et. Al., “GLUE: A Multi-Task Benchmark and Analy-sis Platform for Natural Language Understanding“, International Conference on Learning Representations, ICLR 2019.