트랜스포머 디코더의 토큰 가지치기를 위한 누적 어텐션 스코어 감쇄 기법

조현래[0], 신동군
성균관대학교 전자전기컴퓨터공학과
hrae10324@gmail.com, dongkun@skku.edu

Accumulated Attention Score Decaying Technique for Transformer Decoder Token Pruning

Hyun-rae Jo[0], Dongkun Shin
Sungkyunkwan University Department of Electrical and Computer Engineering

요 약

대규모 언어 모델들의 등장으로 인해 성능을 유지하며 모델을 경량화 할 수 있는 방법에 대한 요구가 늘어나고 있다. 이에 누적 어텐션 스코어를 기반으로 한 토큰 가지치기 기법이 제시되었고, 높은 성능을 이루었다. 그러나 최근 각광받고 있는 디코더 모델들은 그 특성으로 인해 어텐션 스코어를 누적하는 과정에서 토큰간 불균형이 발생한다. 본 논문에서는 이를 해결할 수 있도록 누적 어텐션 스코어 기반 토큰 가지치기에 감쇄 인자를 도입할 것을 제안한다. 이를 통해 기존 기법 대비 OPT-2.7B 모델에서 4.0%의 정확도 향상을 얻을 수 있었다.

1. 서 론

최근 대규모 언어 모델(Large Language Model, LLM)이 높은 성능을 보이며 다양한 산업에 영향을 미치고 있다. 그러나 동시에 모델의 크기가 방대해지고 필요한 메모리도 증가하는 문제가 발생하고 있다. 이 문제를 해결하기 위한 방법 중 하나로, 연산 중 불필요한 토큰을 제거하는 토큰 가지치기가 제안되었다. 토큰 가지치기는 다양한 방법을 활용하여 각 토큰의 중요도를 점수화하고, 점수가 낮은 토큰을 제거함으로써 연산량과 메모리 사용량을 줄인다.

대표적인 중요도 측정 방법 중 하나는 SpAtten[1]이다. SpAtten은 Self-Attention 연산 중 발생하는 어텐션 스코어를 토큰 단위로 누적하여 중요도를 판단한다. 또한, SpAtten에서 모든 레이어에 똑같은 토큰 가지치기가 적용되는 것을 개선하여, 레이어와 헤드 단위로 중요도를 측정하고, 가지치기 하는 토큰을 다르게 설정하는 H2O[2] 기법도 제시되었다.

한편, 트랜스포머 디코더[3]는 자신 이후의 토큰들에 대해서는 연산을 진행하지 않는 Masked Self-Attention을 사용하고 있다. 이 과정에서 [그림 1(b)]와 같은 마스크가 도입되며, 이 마스크가 적용된 어텐션 스코어에 위의 축적 기법을 그대로 적용하면, 등장 순서로 인한 토큰들 간의 불균형이 발생한다. 이 불균형은 문장 앞쪽의 토큰이 선택될 확률을 높이고 토큰들 간의 공평한 비교를 불가능하게 한다. 따라서, 오랫동안 축적된 토큰에는 그에 맞는 페널티를 부과하여서 토큰 간의 불균형을 해소해야 한다

* 이 논문은 2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.IITP-2017-0-00914, 지능형 IoT 장치용 소프트웨어 프레임워크)

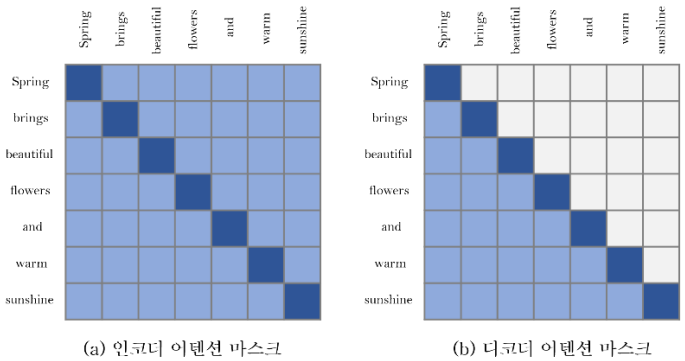
2. 배 경

2.1. 디코더 기반 모델의 특성

GPT나 LLaMA와 같은 트랜스포머 디코더 기반 모델은 자동 회귀(Auto-regressive, AR) 특성을 활용한다.

자동 회귀 모델은 시퀀스를 생성할 때 이전에 생성된 토큰들을 현재 시점의 입력으로 사용한다. 예를 들어, [그림 1(b)]에서 "Spring brings beautiful"이라는 토큰들을 입력으로 사용하여 "flowers"라는 출력을 생성하고, 이를 추가한 "Spring brings beautiful flowers"가 다음 입력으로 사용된다.

따라서 실제적으로 사용되는 부분은 어텐션 연산의 하삼각행렬[그림 1(b)]에 해당하며, 인코더 모델[그림 1(a)]과는 다르게 상삼각행렬은 값이 없어지게 된다. 이는 모델이 현재 시점에서 미래의 정보에 접근할 수 없음을 의미한다. 이러한 특성은 모델이 실제 사용 시나리오에서 미래의 정보에 접근할 수 없는 것을 반영한 것이다.



(a) 인코더 어텐션 마스크 (b) 디코더 어텐션 마스크

[그림 1] 인코더와 디코더의 어텐션 마스크

2.2 토큰 가지치기

어텐션 연산의 기본 구조는 다음과 같다:

$$Q = W_Q \cdot X, \quad K = W_K \cdot X, \quad V = W_V \cdot X \quad \cdots (1)$$

$$\text{Score} = \text{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d}}\right) \quad \cdots (2)$$

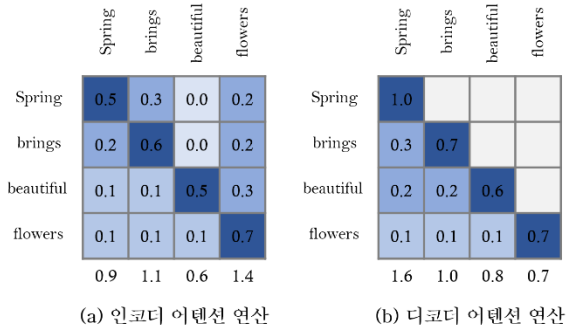
$$\text{Attention} = \text{Score} \cdot V \quad \cdots (3)$$

여기서, W_Q, W_K, W_V는 각각 Q, K, V에 대한 가중치를 나타내며, X는 입력, d는 모델의 중간 차원(hidden dimension)을 의미한다. 현재 시퀀스의 길이를 N이라고 할 때, Q, K, V는 (N, d)의 차원을 가진다. (식 2) 연산의 $Q \cdot K^T$는 $N \times d \times N = O(N^2)$로, N의 제곱에 비례하는 연산량을 가지게 된다.

한편, 자동 회귀 과정에서는 매번 모든 시퀀스에 대한 Q, K, V를 계산하는 것이 아니라, 이전 시퀀스에 해당하는 N개의 시퀀스에 대한 K_N와 V_N를 메모리에 저장해두고, 새로 추가된 토큰에 대한 $(1, d)$차원의 K_{new}, V_{new}와 합쳐서 $(N+1, d)$ 차원의 새로운 K_{N+1}과 V_{N+1}을 생성한다. 이 때 저장해두는 N개에 대한 K와 V를 KV 캐시라고 부르며, 이는 N에 비례하여 크기가 커지고 따라서 메모리 사용량이 증가한다.

이러한 문제를 해결하기 위한 한 가지 방법은 토큰 가지치기이다. 토큰 가지치기는 불필요한 토큰을 제거함으로써 위의 N을 줄이는 것을 목표로 한다. 이를 위해 각 토큰들의 중요도를 파악할 필요가 있고, SpAtten에서는 어텐션 스코어(식 2)의 열방향 합을 각 토큰의 중요도로 간주한다. 즉, 누적하여 많은 점수를 생성한 토큰을 중요한 것이라 판단한다. 예를 들어, [그림 2(a)]에서는 "beautiful"이 가장 불필요한 토큰으로, "flowers"가 가장 중요한 토큰으로 판단된다. 한편, H2O에서는 모든 레이어와 헤드에 같은 가지치기를 적용하는 SpAtten을 개선하여 레이어와 헤드별로 누적 점수를 각각 유지하고, 서로 다른 가지치기를 적용하는 것을 제시하였다.

그러나 이 기법을 트랜스포머 디코더 모델에 적용하면 문제가 발생한다. [그림 2(b)]에서, "Spring"에는 총 4개의 값이 축적되고 "flowers"에는 1개의 값만 축적된다. 따라서 더 많은 값이 축적된 앞쪽의 토큰의 중요도가 높아진다. 예시에서도 등장 순서대로 중요도가 설정되는 문제가 나타난다. 따라서 본 논문에서는 이 문제를 해결하기 위해 어텐션 스코어를 누적하는 과정에 감쇄 인자를 도입할 것을 제안한다.



(a) 인코더 어텐션 연산 (b) 디코더 어텐션 연산

[그림 2] 인코더와 디코더에서의 누적 어텐션 스코어

3. 본 문

3.1. 감쇄 인자

기존의 i번째 토큰의 N번째 시퀀스에서의 누적 어텐션 스코어 계산 방식은 아래와 같다:

$$S_{i,N} = \text{Score}_{i,N} + S_{i,N-1} \quad \cdots (4)$$

본 논문에서는 위의 식에 감쇄 인자 α를 도입하여 아래와 같이 변경할 것을 제안한다:

$$S_{i,N} = \text{Score}_{i,N} + \alpha \cdot S_{i,N-1} \quad \cdots (5)$$

$$0 < \alpha < 1$$

위와 같이 변경함으로써 과거에 발생했던 점수에는 α가 반복적으로 곱해져서 그 크기가 작아진다. 따라서 다수의 값이 누적되더라도 그만큼 감쇄 정도가 커지기 때문에 누적이 많이 되지 않은 토큰들과의 불균형이 해소된다.

또한, α의 값을 조절하는 것을 통해 과거에 두는 비중을 조절할 수 있다. α에 1과 가까운 값을 할당할 경우 과거에 발생한 큰 점수가 현재에도 유지되기 때문에 현재에 영향을 미칠 수 있으나, α가 0에 가까울 경우 빠르게 점수가 작아지기 때문에 최근의 결과만을 사용하여 토큰들을 비교할 수 있다.

4. 실험결과

4.1. 실험환경

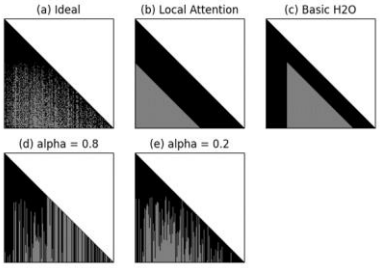
본 논문에서는 OPT-2.7B[4] 모델을 사용하였고, OpenbookQA, Winogrande, ARC-e, ARC-c, PiQA 데이터세트에서 0-shot 성능을 측정하였다. 실험결과에서 "Full"은 토큰 가지치기를 적용하지 않은 경우, "Local"은 Local Attention을 사용한 경우, "H2O"는 감쇄 인자를 적용하지 않은 기존 H2O의 가지치기를 적용한 경우, "Decay"는 논문에서 제시한 감쇄 인자를 적용한 경우이다. Local, H2O, Penalty 모두 Full 조건의 40%에 해당하는 토큰만을 남겼다.

4.2. 실험결과

4.2.1. 마스크 패턴

[그림 3]에서 관찰할 수 있는 검은색 영역은 토큰 가지치기 과정에서 선택된 토큰들을 나타낸다. [그림 3(a)]는 토큰을 제거하지 않고 계산하여 각 행마다 어텐션 스코어가 높은 것들을 선택한 이상적인 마스크이며, [그림 3(b)]는 Local Attention 기법으로, 주어진 길이만큼의 최신 토큰을 남긴다. [그림 3(c)]는 기존의 H2O 기법을 적용한 결과로, 초기 토큰들만 선택되는 문제가 발생함을 보여준다. 이러한 문제를 보완하기 위해, 기존의 H2O 기법은 Local Attention과 결합되어 사용되며, Local Attention이 없을 경우 성능 저하가 발생한다. Local Attention과 H2O를 통해 선택되는 토큰의 비율은 [표 1]과 같다.

반면, [그림 3(d), (e)]는 α=0.2, 0.8을 적용한 결과를 보여준다. 이 방법은 기존의 방법과 달리 모든 시점에 대해 균형있게 토큰을 선택함을 확인할 수 있다. 특히, Local Attention이 필요한 경우와 필요하지 않은 경우를 자체적으로 고려하여 다양한 형태의 마스크가 생성된다. 따라서 본 논문의 기법은 Local Attention에 토큰을 할당하지 않더라도 필요한 부분에서 그 특징을 볼 수 있다. 또한, α가 작을 경우 최신에, 클 경우 과거에 중점을 두는 모습을 확인할 수 있으며 기존 H2O 대비 [그림 3(a)]에 가까운 마스크를 만드는 것을 볼 수 있다.



[그림 3] 가지치기 마스크

기법	Local	Select
Full	100%	
H2O	20%	20%
Decay	0%	40%

[표 1] 토큰 비율

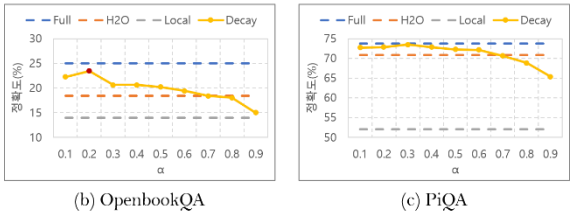
4.2.2. 일반 추론(Commonsense Reasoning)

[그림 4(a)]에서 H2O 를 적용했을 때, Full 토큰 대비 평균 5.9%의 정확도가 하락하는 것을 확인할 수 있다. 반면, 감쇄를 적용한 논문의 방법을 사용하였을 때에는 4.0%의 정확도가 상승하여 가치지기를 적용하지 않았을 때와 1.9%만의 차이를 보였다.

또한 [그림 4(b), (c)]에서와 같이 데이터 세트마다 적절한 α 가 다를 수 있음을 확인하였고, 따라서 사용하고자 하는 환경에 맞는 α 를 설정해줌으로써 최대한의 성능을 낼 수 있는 것을 확인할 수 있었다.

기법	OpenbookQA	Winogrande	ARC-e	ARC-c	PiQA	Average
Full	25.0	60.9	60.8	26.9	73.8	49.5
H2O	18.4	55.0	50.7	23.0	70.9	43.6
Decay	**23.4**	**55.6**	**60.1**	**26.1**	**72.8**	**47.6**

(a) 전체 실험 결과: α=0.2



(b) OpenbookQA (c) PiQA

[그림 4] OPT-2.7B 실험 결과

4.2.3. LLaMA 2-7.0B(Chat)에서의 성능

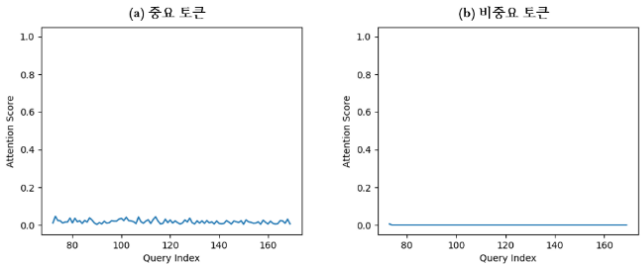
OPT-2.7B 모델보다 더 큰 규모의 LLaMA 2-7.0B(Chat)에서는 4.6%의 정확도 향상을 보였다.

기법	OpenbookQA	Winogrande	ARC-e	ARC-c	PiQA	Average
Full	33.2	66.5	73.9	44.2	76.4	58.8
H2O	24.8	55.4	64.1	36.6	73.4	50.9
Decay	**27.4**	**59.0**	**72.9**	**42.9**	**75.1**	**55.5**

[표 2] LLaMA 2-7.0B(Chat) 실험 결과

4.2.4. 감쇄 인자의 역할

[그림 4(a), (b)]에서 α의 값이 작을 때의 정확도가 α의 값이 클 때보다 높은 경향을 보이고 있다. 이는 중요 토큰과 비중요 토큰 간의 어텐션 스코어의 크기 차이가 미미하기 때문이다([그림 5(a), (b)]). 따라서, α 값이 높아 과거를 과도하게 고려하는 경우, 과거에 기반한 점수가 중요 토큰의 점수를 초과하는 문제가 발생하게 된다. 한편, [그림 5(a)]와 같이 중요 토큰의 점수가 일정하게 크지 않기 때문에, 낮은 α 값을 통해 짧은 과거를 고려함으로써 일시적인 점수 저하를 보완하는 것이 토큰 선택에 더 유리한 결과를 야기한다.



[그림 5] 중요, 비중요 토큰의 어텐션 스코어

5. 결 론

본 논문에서는 누적 어텐션 스코어 기반 토큰 가지치기에 감쇄 인자를 도입할 것을 제안한다.

트랜스포머 디코더 모델의 특성으로 인해 발생하는 누적 어텐션 스코어 간의 불균형을 해소하여 토큰 가지치기 과정에서 앞쪽 토큰이 주로 선택되는 현상을 개선할 수 있으며, Local Attention 에 대한 필요성을 제거할 수 있었다. 이를 통해 기존 대비 정확도 향상을 달성할 수 있었다.

실험을 통해 기존 H2O 기법 대비 OPT-2.7B 모델에서 4%, LLaMA 2-7.0B(Chat) 모델에서 4.6%의 정확도 향상을 확인하였으며, 적용 분야에 따른 α값을 조정하는 것을 통해 성능을 최대화할 수 있는 것을 확인하였다.

참 고 문 헌

[1] Hanrui, W., et al. Spatten: Efficient sparse attention architecture with cascade token and head pruning. IEEE International Symposium on High Performance Computer Architecture (HPCA), pages 97-110. IEEE, 2021.

[2] Zhenyu Z., et al. H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. arXiv preprint arXiv: 2306.14048(2023)

[3] Ashish V., et al. Attention is all you need. In NIPS, 2017.

[4] Susan Z., et al. Opt: Open pre-trained transformer language models. arXiv preprint arXiv:2205.01068, 2022.