Shared Attention Network와 Multi-Query Attention을 이용한 Transformer 모델의 디코더 속도 개선

최민주 ○, 김도경, 황현선, 이창기, 이성민, 류우종, 염홍선, 김준석
강원대학교 빅데이터메디컬융합학과, 현대자동차
{mjchoi0831, ehrud235}@gmail.com, {hhs4322, leeck}@kangwon.ac.kr,
{ blueworm7, woojong.ryu, yeom.j.hs, junseok.kim}@ hyundai.com

Speed Improvement of Transformer Model using Shared Attention Network and Multi-Query Attention

Minjoo Choi○, Dokyoung Kim, Hyunsun Hwang, Changki Lee,
Seongmin Lee, Woojong Ryu, Hongseon Yeom, Junseok Kim
Dept. Medical Bigdata Convergence, Kangwon National University
Hyundai Motor Company AIR Lab

요 약

신경망을 이용하는 기계번역 (Neural Machine Translation) 모델 중에 Transformer를 이용하는 기계번역 모델의 경우 우수한 번역 성능을 보이고 있으나 메모리 사용량이 많으며 번역 속도가 느리다는 단점이 있다. 이러한 단점을 보완하기 위해 본 논문에서는 Transformer의 디코더 구조를 개선하여 디코더의 속도를 향상시켰다. 선행 연구를 바탕으로 전체 레이어 수를 줄이고 디코더 안팎으로 Attention Mechanism에 필요한 요소를 공유하는 방법을 통해 연산 횟수 및 연산에 필요한 텐서 크기를 줄인 결과 번역 속도가 향상되었다.

주제어: 기계번역, NMT, Transformer, Shared Attention Networks, Multi-Query Attention

1. 서론

최근 신경망을 이용한 기계번역(Neural Machine Translation) 연구들이 다양하게 진행되고 있다. 다양한 모델 중 Transformer[1] 모델이 가장 우수한 번역 성능을 보이나 모델 구조가 복잡하여 메모리 사용량이 많으며 번역 속도가 느리다는 단점이 있다. 이에 따라 신경망 기계번역 모델의 메모리 사용량을 줄이고 번역 속도를 증가시키기 위한 기술이 활발히 연구되고 있다.

본 논문에서는 번역 속도를 증가시키기 위해 Transformer의 디코더 구조를 개선하여 디코더의 속도를 향상시켰다. 우선 인코딩보다 디코딩 시간이 더 오래 걸리므로 디코더 레이어의 갯수를 줄여 디코딩 시 총 연산 횟수를 대폭 줄였고, 디코더에서 인접한 레이어 간의 Attention 값을 공유하여 Attention에 필요한 연산 횟수를 줄였다. 추가적으로 디코더 레이어의 Multi-head Attention 과정에서 Attention Head를 공유하여 연산에 필요한 Tensor 크기를 줄인 결과 메모리 대역폭 요구량이 감소하여 연산 속도가 향상되었다.

2. 관련 연구

기존의 Transformer 모델을 변형하여 디코더의 속도를 개선시키는 여러 연구가 있었다.

[2]의 연구에서 AR(Auto Regressive)과 NAR(Non-Auto Regressive) 모델 모두 인코더보다 디코더에서 계산 시간이 더 많이 소요되므로 인코더 레이어의 갯수는 늘리거나 그대로 유지하고, 디코더 레이어의 갯수를 줄이면 모델의 번역 속도가 증가함을 입증하였다. 단, BLEU Score와 속도가 반비례, 즉 Trade-off 되는 경향이 있기 때문에 본 논문에서는 BLEU Score를 고려하여 인코더 레이어 갯수는 4개, 디코더 레이어 갯수는 2개로 각각 축소하였다.

[3]에서는 디코더 연산 속도를 향상시킬 수 있는 SAN(Shared Attention Networks) 구조를 제안하였다. 인접한 레이어 간의 Attention Weight가 유사한 점에 착안하여 인접한 레이어끼리 Attention 값을 공유하도록 설계하였다. 디코더에서 Self-Attention Weight 값과 Encoder-Decoder Attention 값을 인접한 레이어끼리 공유한 결과 연산 횟수가 줄어든 만큼 속도가 향상되었다.

[4]에서는 Multi-head Attention 구조를 변형한 MQA (Multi-query Attention)를 제안하였다. Multi-head Attention 과정에서 Key와 Value가 각각 하나의 Attention Head를 공유하여 Tensor 크기를 감소시키고, 이는 메모리 대역폭 요구량 감소로 이어져 모델의 연산 속도가 향상되는 결과를 얻었다.

본 논문에서는 Transformer의 디코더의 속도를 향상시키기 위해 SAN 모델과 MQA를 결합한 모델을 제안하고, 한국어-영어 기계번역 모델에 적용하였다.

이 논문은 현대 자동차 AIR Lab의 "신경망 기계번역 모델을 위한 지식 증류 기술 연구" 과제의 지원을 받아 수행된 연구임

3. 개선 모델 구조

본 논문에서는 Transformer 모델의 인코더 레이어 갯수를 4개로 유지하고, 디코더 레이어 갯수를 2개로 축소하여 전체 연산량을 대폭 줄였다.

이어서 디코더 측에서 SAN 모델과 같이 인접한 레이어 간의 Self-attention Weight 값 및 Encoder-Decoder attention 값을 공유하도록 하였다. 그림 1은 SAN 모델의 디코더에서 Attention 값이 다음 레이어로 공유되는 과정을 나타낸다.

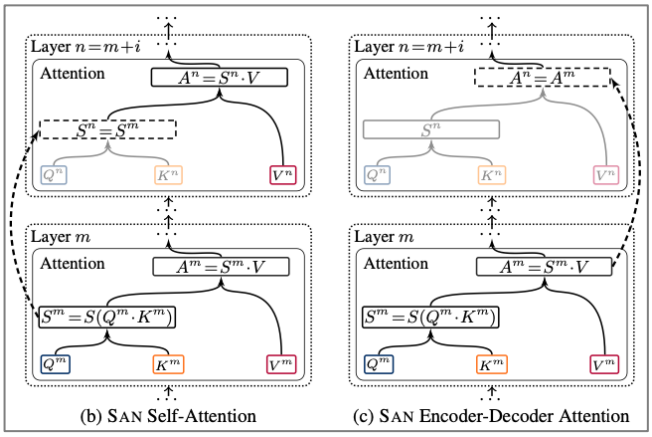

그림 1 SAN Attention 구조 [3]

(b)는 SAN 모델의 디코더 레이어 간에 Self-attention Weight를 공유하는 과정으로, m번째 레이어에서 $S^m = S(Q^m, K^m)$ 연산 결과로 얻은 Self-attention Weight Matrix인 S^m을 $n(=m+i)$번째 레이어의 S^n으로 전달한다. 다시 말해 이전 레이어의 Self-attention Weight를 다음 레이어로 공유하여 각각의 레이어 마다 Self-attention Weight 값을 계산하지 않도록 한다.

(c)는 SAN 모델의 디코더 레이어 간에 Encoder-Decoder Attention의 결과 값을 공유하는 과정이다. $A^m = S^m \cdot V$ 연산을 통해 얻은 m 번째 레이어의 Encoder-Decoder Attention 결과 값 A^m을 n번째 레이어의 A^n으로 전달한다. 즉 이전 레이어의 Encoder-Decoder Attention 값을 다음 레이어의 Attention 값으로 공유하여 각각의 레이어에서 Encoder-Decoder Attention을 계산하지 않도록 한다.

추가적으로 본 논문에서는 MQA 모델을 SAN과 결합하여 디코더의 Multi-head Attention 모듈에서 Key와 Value의 Attention Head를 각각 공유하도록 변경하였다. 이와 같이 디코더에서 인접한 레이어 간의 Attention 값을 공유하되 각각의 레이어 내부에서는 Key와 Value의 Attention Head를 공유하여 레이어 안팎으로 연산에 필요한 값을 공유한 결과 모델 크기 및 파라미터 수가 감소하였고, 연산 횟수가 줄어든 만큼 번역 속도가 향상되는 효과를 얻었다.

4. 실험 및 결과

4.1 실험 설정

본 논문에서는 OpenNMT[5] 오픈소스 프레임워크의 Transformer 모델을 실험 목적에 맞게 수정하여 사용하였다. 학습을 위한 GPU로 Nvidia GeForce GTX 1080을 이용하였으며, dropout 0.1, label smoothing 0.1, learning rate 2.0, beam size 1을 사용하였고, 최적화 함수로 LazyAdam 을 사용하였다. 학습 도중 과적합을 방지하기 위해 Early Stop을 5로 설정하였으며 성능 측정 지표로 BLEU[6]를 사용하였다.

비교를 위해 총 4개의 모델을 실험하였다. 우선 레이어 갯수에 따른 성능을 비교하기 위해 (1) 인코더-디코더 레이어 갯수가 모두 6개인 Transformer 기본 모델(Baseline)과 (2) 인코더-디코더 레이어 갯수를 각각 4개, 2개로 줄인 Transformer 기본 모델에 대해 실험하였다. 이어서 (2) 와 동일한 레이어 갯수를 가진 각각 다른 구조의 모델을 비교하기 위해 인접한 레이어 간의 attention 값을 공유하는 (3) SAN 모델과, SAN 모델에 MQA를 접목시킨 (4) SAN+MQA 모델에 대해 실험하였다.

또한 임베딩 벡터와 레이어의 차원 수를 줄여 모델을 경량화하는 경우의 성능 및 속도를 비교하기 위해 차원 수가 512, 256, 128인 경우에 대해 각각 실험하였다.

4.2 평가 및 결과

평가를 위해 AI Hub[1] 에서 제공하는 한국어-영어 번역(병렬) 말뭉치 데이터를 사용하였다. 모델 학습에 798,187 쌍, 테스트에 1,600 쌍의 병렬 문장 쌍을 각각 이용하였다.

모델의 번역 속도를 확인하기 위해 Intel Core i9-9900K CPU로 수행 시간을 측정하였다. 표 1은 각 모델의 조건에 따른 모델의 크기, 번역 성능, 번역하는 데 걸리는 시간을 측정한 결과로 레이어 수와 임베딩 차원 수를 줄일수록 모델의 크기 및 파라미터가 감소하여 모델이 경량화되는 만큼 BLEU Score는 소폭 감소하지만 번역 속도가 증가함을 보인다. 또한 SAN 모델과 본 논문에서 제안한 SAN+MQA 모델이 Attention 값을 공유하여 번역 속도가 증가함을 알 수 있다.

결과적으로 같은 조건일 경우 본 논문에서 제안한 SAN 구조를 베이스로 하여 Multi-head Attention을 Multi-query Attention으로 대체한 SAN+MQA 모델의 번역 속도가 가장 빠르며, 이는 제안한 모델이 메모리를 적게 차지하면서도 번역 속도가 빠르므로 저용량 디바이스에서 실시간으로 번역이 이루어지는 시스템에서 효과적으로 활용될 수 있음을 의미한다.

[1] https://aihub.or.kr/

표 1 모델 조건에 따른 성능 및 수행시간 측정 결과

	Layers	d_{model}	d_{FF}	#heads	Model size (MB)	#Parameters	BLEU	Total prediction time (s)	Average prediction time (s)	Tokens per second
Transformer	6, 6	512	2048	8	921.9	76,808,214	39.39	95.79	0.0599	574.84
	4, 2				644.3	53,687,318	36.99	54.89	0.0343	985.44
SAN	4, 2				634.8	52,899,350	36.48	51.72	0.0323	1039.17
SAN + MQA	4, 2				**618.3**	**51,520,406**	34.89	**49.24**	**0.0308**	**1096.11**
Transformer	6, 6	256	512	4	291.3	24,258,070	37.28	46.09	0.0288	1191.54
	4, 2				240.5	20,040,726	33.71	25.80	0.0161	2067.17
SAN	4, 2				238.2	19,843,350	30.81	26.68	0.0167	2008.86
SAN + MQA	4, 2				**234.6**	**19,547,286**	30.87	**23.00**	**0.0144**	**2307.83**
Transformer	6, 6	128	256	2	122.3	10,178,838	31.77	30.14	0.0188	1805.13
	4, 2				109.5	9,118,742	27.00	19.12	0.0120	2732.43
SAN	4, 2				108.9	9,069,206	24.58	19.23	0.0120	2737.15
SAN + MQA	4, 2				**108.3**	**9,019,670**	24.37	**18.72**	**0.0117**	**2852.49**

5. 결론

신경망을 이용하는 기계번역 (Neural Machine Translation) 모델은 우수한 번역 성능을 보이나 메모리 사용량이 많으며 번역 속도가 느리다는 단점이 있다. 이러한 단점을 보완하기 위해 본 논문에서는 Transformer 모델의 디코더 구조를 수정하여 속도를 개선하였다.

모델의 크기 및 연산 횟수를 줄이기 위해 인코딩보다 디코딩 시간이 더 오래 걸리므로 디코더 레이어의 갯수를 최소화하고 레이어 안팎으로 연산에 필요한 값을 공유하도록 디코더 구조를 수정하였다. 디코더에서 인접한 레이어 간의 Self-attention Weight 값 및 Encoder-Decoder Attention 값을 공유하여 Attention 에 필요한 연산 횟수를 줄여 속도를 끌어올렸으며, 디코더 레이어 내부의 Multi-head Attention 모듈의 Key와 Value가 각각 하나의 Attention Head를 공유하도록 수정하여 연산에 필요한 텐서 크기를 줄인 결과 메모리 대역폭 요구량이 감소하여 연산 속도가 향상되었다.

본 연구는 기존의 신경망 기계번역 모델에 비해 메모리 사용량이 적고 빠르기 때문에 모바일 환경과 같은 저사양 디바이스에서 실시간 번역에 활용될 수 있다.

참고문헌

[1]Vaswani, Ashish, et al. "Attention is all you need." arXiv preprint arXiv:1706.03762, 2017.

[2]KASAI, Jungo, et al. Deep encoder, shallow decoder: Reevaluating the speed-quality tradeoff in machine translation. arXiv preprint arXiv:2006.10369, 2020.

[3]XIAO, Tong, et al. Sharing attention weights for fast transformer. arXiv preprint arXiv:1906.11024, 2019.

[4]SHAZEER, Noam. Fast transformer decoding: One write-head is all you need. arXiv preprint arXiv:1911.02150, 2019.

[5] KLEIN, Guillaume, et al. Opennmt: Open-source toolkit for neural machine translation. arXiv preprint arXiv:1701.02810, 2017.

[6]Papineni Kishore, et al. BLEU: a method for automatic evaluation of machine translation. Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics. 2002.