LoRA: Low-Rank Adaptation of Large Language Models

Abstract&introduction

- NLP는 LLM을 다운스트림에 맞게 fine-tuning하는게 일반적
- 모델이 크면 클수록 풀파인튜닝이 비효율적

이에 따라, `LoRA→저차원 행렬을 학습하고 모델 가중치는 동결` 하는 방식 제시

그에 따라, 다음과 같은 장점이 있음

- 학습 파라미터 10000배 감소
- GPU 사용량 3배 절약
- 추론 속도 저하 x

Terminologies

- dmodel: Transformer 레이어의 입력 및 출력 차원 크기
- Wq, Wk, Wv, Wo: self-attention 모듈에서 query, key, value, output projection 행렬
- W 또는 W0: 사전 학습된 가중치 행렬
- ΔW: Adaptation 중에 누적된 기울기 업데이트
- r: LoRA 모듈의 rank

Method

1. Low-Rank Parametrized Update Matrices

- 기존 신경망의 **가중치 행렬을 직접 업데이트하는 대신, 저차원 행렬(A, B)로 가중치 변화를 표현**.

사전 훈련된 가중치 W_0를 그대로 유지하면서, 변화량 ΔW를 BA로 근사.

$$W_{\text{new}} = W_0 + \Delta W = W_0 + BA$$

- 여기서:

    - B는 d×r 행렬 (초기값 0)
    - A는 r×k 행렬 (정규분포 N(0,σ2)로 초기화)
    - **r≪min(d,k)** (즉, 저차원 근사)
- 학습 중 W0는 고정되고, **A와 B만 학습**.
- **추론 시 추가적인 연산이 필요하지 않으며, 기존 모델과 동일한 속도로 동작**.

2. Transformer에서 LoRA 적용

- Transformer의 **Self-Attention 모듈의 가중치 행렬 Wq,Wk,Wv,Wo에 LoRA를 적용**.
- 실험적으로 **MLP 모듈은 고정하는 것이 더 효과적**이었음.
- 학습 가능한 파라미터 수를 줄이면서도 **성능 저하 없이 fine-tuning과 유사한 성능을 달성**.

3. LoRA의 주요 장점

1. **메모리 & 계산량 절감**
    - 기존 Fine-Tuning 대비 **학습 가능한 파라미터 수 10,000배 감소**.
    - GPT-3 175B 모델의 **VRAM 사용량을 1.2TB → 350GB로 감소**.
    - Adam 옵티마이저를 사용할 때 **메모리 사용량 3배 절감**.
2. **추론(Inference) 속도 저하 없음**
    - 학습된 BA를 사전 훈련된 W0에 합쳐서 저장하므로, **추론 시 추가 연산 없음**.

      BABA

      W0W_0
    - 즉, 기존 Fine-Tuning 모델과 동일한 속도로 실행 가능.
3. **다양한 Task 전환 용이**
    - 하나의 **사전 훈련된 모델을 공유**하면서, **작은 LoRA 모듈(A, B)만 교체하여 다양한 작업 수행 가능**.
    - 이를 통해 **저장 공간 절약 및 빠른 전환 가능**.

Results(실험 결과)

1. GLUE 벤치마크 (NLP 모델 평가)

- **비교 모델**: RoBERTa, DeBERTa
- **비교 기법**: Full Fine-Tuning (FT), BitFit, Adapter, LoRA 등
- **결과**:
  - **LoRA는 Full Fine-Tuning과 유사한 성능을 달성**하면서도 **훨씬 적은 파라미터만 학습**.
  - RoBERTa-base에서 **LoRA는 0.3M 파라미터로 FT(125M)와 유사한 성능**을 기록.

2. GPT-2 기반 NLG (자연어 생성)

- **비교 모델**: GPT-2 M, GPT-2 L
- **비교 기법**: Full Fine-Tuning, Adapter, PreLayer, LoRA
- **평가 지표**: BLEU, NIST, METEOR, ROUGE-L, CIDEr
- **결과**:
  - **LoRA는 모든 지표에서 기존 Fine-Tuning과 유사하거나 더 나은 성능**을 보임.
  - 특히 **ROUGE-L과 CIDEr 점수가 가장 높음**, 즉 문장 생성 품질이 뛰어남.

3. GPT-3 기반 Task 성능 비교

- **비교 모델**: GPT-3 175B
- **비교 기법**: Full Fine-Tuning, BitFit, Adapter, PreEmbed, PreLayer, LoRA
- **평가 데이터셋**: WikiSQL, MNLI, SAMSum
- **결과**:
  - **WikiSQL**: LoRA(74.0%)가 Full Fine-Tuning(73.8%)보다 높음.
  - **MNLI**: LoRA(91.7%)가 Adapter-H(91.5%)보다 성능 우수.
  - **SAMSum (요약)**: LoRA가 가장 높은 요약 성능(53.8/29.8/45.9) 기록.

4. LoRA 적용 시 학습 가능한 파라미터와 성능 관계

- **Fine-Tuning 대비 10,000배 적은 파라미터로도 유사한 성능 달성**.
- **GLUE 벤치마크와 WikiSQL 데이터셋에서 LoRA의 성능이 안정적으로 유지됨**.

5. Transformer 가중치 행렬에서 LoRA 적용 위치 분석

- Transformer에서 **Wq,Wk,Wv,Wo 중 어느 행렬에 LoRA를 적용해야 하는지 실험**.

- 결과적으로 **Query (Wq)와 Key (Wk) 행렬에 적용하는 것이 가장 효과적**.

6. LoRA의 최적 Rank (r) 분석

- 다양한 **Rank (r) 값에 대한 실험 수행**.
- **WikiSQL과 MultiNLI에서 r=4가 가장 성능이 우수**.
- **높은 Rank는 불필요하며, 작은 Rank만으로 충분한 표현력 제공**.

7. LoRA 업데이트 행렬 (ΔW\Delta WΔW)의 Rank 분석

- **Low-rank 구조가 실제로 효과적인지 검증**.
- 실험 결과, **LoRA가 사전 훈련된 가중치와 높은 상관관계를 유지하면서도 적은 Rank로도 충분한 성능을 보임**.