LoRA: Low-Rank Adaptation of Large Language Models 논문 정리

Abstract (초록)

핵심 아이디어

- 대규모 사전학습 모델 (Pretrained LLM)을 특정 작업에 적응시키는 기존 방식(Full Fine-tuning)의 한계 해결
- GPT-3 175B 같은 초대형 모델의 파인튜닝은 비용적으로 비현실적
- **Low-Rank Adaptation(LoRA)** 제안: **기존 모델 가중치는 고정하고 작은 저랭크 행렬만 학습**
  - 사전학습된 모델의 가중치는 **동결(freeze)**
  - 대신에 각 Transformer 레이어에 **저랭크(rank-r) 행렬(A, B)** 을 삽입하여 업데이트를 re-parametrize

주요 성과

- 학습 파라미터 수 10,000배 감소
- 학습 속도 ↑, 추론 지연 X
- RoBERTa, DeBERTa, GPT-2, GPT-3에서 기존 방식과 동등 또는 더 나은 성능

Introduction

문제 정의

- 대규모 언어모델의 파인튜닝 = 모든 파라미터 재학습 필요
- 모델 크기 증가 → 파인튜닝 비용 기하급수적 증가
- 각 작업별로 전체 모델 크기만큼의 저장공간 필요

기존 해결책의 한계

- **Parameter-efficient 방법들의 문제**

- Adapter layers: 추론 속도 저하
- Prefix tuning: 최적화 어려움, 성능 저하

⇒ Low-Rank Adaptation (LoRA) approach 제안

- Neural network의 layer들을 간접적으로 학습
- rank decomposition matrices (rank로 분해한 행렬들)의 최적화

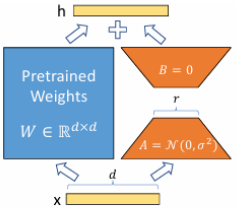


Figure 1: Our reparametrization. We only train A and B.

Problem Statement

수학적 문제 정의

- full finetuning 목표

$$max_{\Phi} \sum_{(x,y) \in Z} \sum_{t=1}^{|y|} log P_{\Phi}(y_t \mid x, y_{<t})$$

- 파인튜닝 시 학습해야 할 파라미터 변화량: $\Delta\Phi$
- $\Delta\Phi$의 크기 ≈ 원본 모델 파라미터 크기
- **목표**: $\Delta\Phi$를 훨씬 작은 파라미터 세트 Θ로 표현 ($|\Theta| \ll |\Phi_0|$)

$$max_{\Theta} \sum_{(x,y) \in Z} \sum_{t=1}^{|y|} log P_{\Phi_0 + \Delta\Phi(\Theta)}(y_t \mid x, y_{<t})$$

현실적 제약사항

- 메모리 사용량
- 저장공간 요구량
- 배포 및 운영 비용

Aren't Existing Solutions Good Enough? - 기존 방법의 한계

1) Adapter Layers의 문제

- **추론 지연시간 증가**: 실시간 서비스에 부적합
- **배치 처리 효율성 저하**
- **하드웨어 병목 현상**

2) Prefix Tuning의 문제

- **최적화 어려움**: 학습 불안정성
- **시퀀스 길이 감소**: 유효 컨텍스트 길이 제한
- **성능 한계**: 복잡한 작업에서 품질 저하

Our Method: LoRA

4.1 Low-Rank-Parametrized Update Matrices

핵심 아이디어

- **가정**: 파인튜닝 시 가중치 변화량 ΔW는 low-rank 구조를 가짐
- **방법**: $\Delta W = BA (B \in R^(d \times r), A \in R^(r \times k), r \ll d, k)$

> 💡 **수학적 구조**
>
> Forward pass: $h = W_0 x + BAx$
> - W_0: 고정된 사전학습 가중치
> - BA: 학습 가능한 저랭크 업데이트
> - r: 랭크 (매우 작은 값, 1~16)

파라미터 효율성

- **전체 파라미터 대비 학습 파라미터 비율**: r(d+k)/dk
- **예시**: r=4, d=k=4096일 때 → 0.2% 파라미터만 학습

4.2 Applying LoRA to Transformer

적용 대상

- **Self-Attention 모듈만 적용**

    - Query, Key, Value, Output projection (Wq, Wk, Wv, Wo)
- **고정 대상**
    - MLP layers
    - Layer Normalization
    - Bias terms

- **메모리 사용량**: 3배 감소
- **저장공간**: 10,000배 감소
- **학습 속도**: 25% 향상
- **작업 전환**: 실시간 가능

제한사항

- 서로 다른 작업의 LoRA 모듈을 동시에 배치 처리하기 어려움

Empirical Experiments

실험 대상 모델

- **RoBERTa** (Base, Large)
- **DeBERTa XXL** (1.5B)
- **GPT-2** (Medium 355M, Large 774M)
- **GPT-3** (175B)

실험 데이터셋

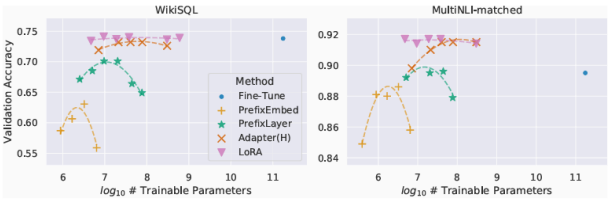
- **자연어 이해**: GLUE benchmark
- **자연어 생성**: E2E NLG Challenge
- **질의응답**: WikiSQL
- **요약**: SAMSum

결과

- LoRA는 full-finetuning**과 동등하거나 우수**

- 특히 GPT-3 175B에서 LoRA가 풀파인튜닝보다 성능 우위
- 파라미터 효율성, 속도, 성능 모두 확보함

Model&Method	# Trainable Parameters	WikiSQL Acc. (%)	MNLI-m Acc. (%)	SAMSum R1/R2/RL
GPT-3 (FT)	175,255.8M	**73.8**	89.5	52.0/28.0/44.5
GPT-3 (BitFit)	14.2M	71.3	91.0	51.3/27.4/43.5
GPT-3 (PreEmbed)	3.2M	63.1	88.6	48.3/24.2/40.5
GPT-3 (PreLayer)	20.2M	70.1	89.5	50.8/27.3/43.5
GPT-3 ($\text{Adapter}^{\text{H}}$)	7.1M	71.9	89.8	53.0/28.9/44.8
GPT-3 ($\text{Adapter}^{\text{H}}$)	40.1M	73.2	**91.5**	53.2/29.0/45.1
GPT-3 (LoRA)	4.7M	73.4	**91.7**	**53.8/29.8/45.9**
GPT-3 (LoRA)	37.7M	**74.0**	**91.6**	53.4/29.2/45.1



Related Works

Transformer 기반 언어모델

- GPT, BERT 이후 파인튜닝이 NLP 성능 향상의 표준

Parameter-efficient methods

- Adapter layers, Prompt tuning 등
- LoRA는 기존 work과 달리 추론 지연 없음

저랭크 구조 연구

- 많은 딥러닝 문제에서 intrinsic low-rank 존재
- LoRA는 이를 **업데이트 행렬**에 적용한 최초의 방식

Understanding the Low-Rank Updates

실증 연구

- 어떤 weight에 LoRA 적용이 효과적인가?

적용 대상	파라미터 수	성능 (WikiSQL)
Wq만	0.5M	70.4%
Wk만	0.5M	69.8%
Wv만	0.5M	69.1%
Wq + Wv	**1.0M**	**73.2%**
모든 attention	2.0M	73.8%

→W_q, W_v 동시 적용 시 가장 성능 좋음

- 최적 랭크 r는?

→ 매우 작은 값(예: r=1,2)도 충분히 경쟁력 있음

Rank (r)	파라미터 수	성능	효율성
r=1	0.25M	70.1%	좋음
r=2	0.5M	72.8%	좋음
r=4	**1.0M**	**73.2%**	**균형**
r=8	2.0M	73.5%	보통
r=16	4.0M	73.7%	낮음

- ΔW와 W 관계
  - 무작위가 아닌, W의 덜 강조된 방향을 증폭
  - LoRA는 **사전학습에서 잠재적으로 학습된 특징을 특정 태스크에 맞게 강화**.

Conclusion and Future Work

핵심 기여

- 10,000배 적은 파라미터로 동등한 성능
- 지연 없이 빠른 작업 전환 가능
- 확장성 - 대규모 모델에 실질적 해결책 제공

실무 적용 가능성

- 비용 절감, 배포 효율성, 제한된 자원으로도 대규모 모델 활용

향후 연구 과제

- LoRA 작동 원리의 더 깊은 분석
- 방법론 결합: 다른 효율화 기법과의 통합
- 자동화: 최적 랭크와 적용 대상 자동 결합