LoRA: Low-Rank Adaptation of Large Language Models

Tags	
pdf	https://arxiv.org/pdf/2106.09685.pdf

Introduction

목표: LLM을 더 효율적으로(더 적은 파라미터 업데이트, 더 적은 메모리 사용) 전이학습(다른 도메인에 적용)하기

- **기존의 full fine-tuning은 모든 파라미터를 업데이트하느라 실현하기 어려웠다.**

  full fine-tuning: 사전 학습(pre-trained) 모델의 모든 파라미터를 업데이트

  상당한 성능 향상을 이끌어냈다.

  언어 모델의 규모가 커짐에 따라 fine-tuning의 난이도가 급격하게 상승하였다.

- **기존의 대안들: 일부 파라미터만 조정하거나 새로운 태스크를 위한 외부 모듈만을 학습시키기**

  부작용

  - inference latency
  - reduction of the model’s usable sequence length
  - trade-off between efficiency and model quality

- ⭐ LoRA의 가설: 모델 미세 조정 과정에서의 가중치 변화는 **low intrinsic rank**를 가질 것이다.
- 실제로 낮은 rank도 전이 학습시키기에 충분했다.

LoRA만의 차별점

- 같은 사전 학습 모델이 태스크별 LoRA 모듈들에 공유되고 사용될 수 있다, 즉 마치 레고처럼 같은 사전 학습 모델에 LoRA 행렬만 바꿔 끼면 되기 때문에 태스크 전환 오버헤드를 상당 수준 감소시킬 수 있다. 또한 이 때 사전 학습 모듈의 파라미터들은 고정되어 필요햔 저장 용량을 줄일 수 있다.
- 대부분의 파라미터를 고정시키고 새로 삽입한 소규모 low-rank 행렬들만 최적화하기 때무에 효율적이고 하드웨어 진입 장벽을 3배나 낮출 수 이쌀.
- 단순 선형 디자인을 적용하여 inference latency가 없다.
- 기존 방법들에 대해 **orthogonal**해서 그것들과 결합될 수 있다. (예시: prefix-tunning)

> prefix-tunning: 모든 파라미터 동결한 상태에서 모델 앞부분, 즉 입력단에 데이터를 추가하는 방식

Problem Statement

LoRA 방법은 언어 모델링이 아닌 다른 태스크에도 적용될 수 있지만 이 논문에서는 트랜스포머 기반 언어 모델링에 적용함.

언어 모델링의 목적: 주어진 프롬프트에 대한 조건부 확률을 최대화하는 파라미터(Φ)를 찾는 것

기존 fine-tuning:

- 목적 함수:

$$\max_{\Phi} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(P_{\Phi}(\underbrace{y_t|x, y_{<t}}_{\text{autoregressive}})\right)$$

x = context sequence y = target sequence
x_i = context token, y_i = target token
요약 task에서 x가 문단, y가 요약본이라고 생각하면 쉽다.

- 사전 학습 파라미터를 다음과 같이 업데이트:

$$\Phi_0 \rightarrow \Phi_0 + \Delta\Phi, \text{ wh } |\Delta\Phi| = |\Phi_0|$$

$|\Delta\Phi| = |\Phi_0|$이기 때문에 대규모 사전 학습 모델의 $|\Phi_0|$가 매우 커짐에 따라서 업데이트에 필요한 gradient 규모가 매우 커진다. gpt3만 해도 175 Billion.

LoRA:

- 목적 함수:

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

- 사전 학습 파라미터를 업데이트 하지 않고 새로운 파라미터를 task specific data에 대해 업데이트 한 후에 합치기:

$$\Delta\Phi = \Delta\Phi(\Theta),\ \text{wh}\ |\Theta| << |\Phi_0|$$

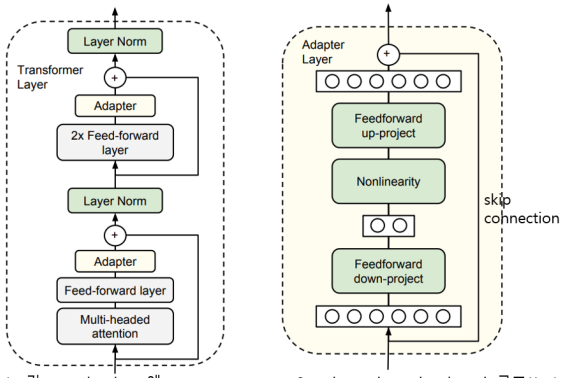
$|\Theta| << |\Phi_0|$이기 때문에 사전 학습 모델의 파라미터를 업데이트하는 것보다 업데이트 시간이 훨씬 단축된다. ($|\Phi_0|$의 0.01%)

Aren’t Existing Solutions Good Enough?

기존의 방식들과 한계:

- **Adding adapter Layers**

  [Parameter-Efficient Transfer Learning for NLP]



1. 각 attention layer에 adapter 모듈을 추가
2. adapter layer: bottleneck 구조(1×1 layer로 파라미터 수 감축)

한계:

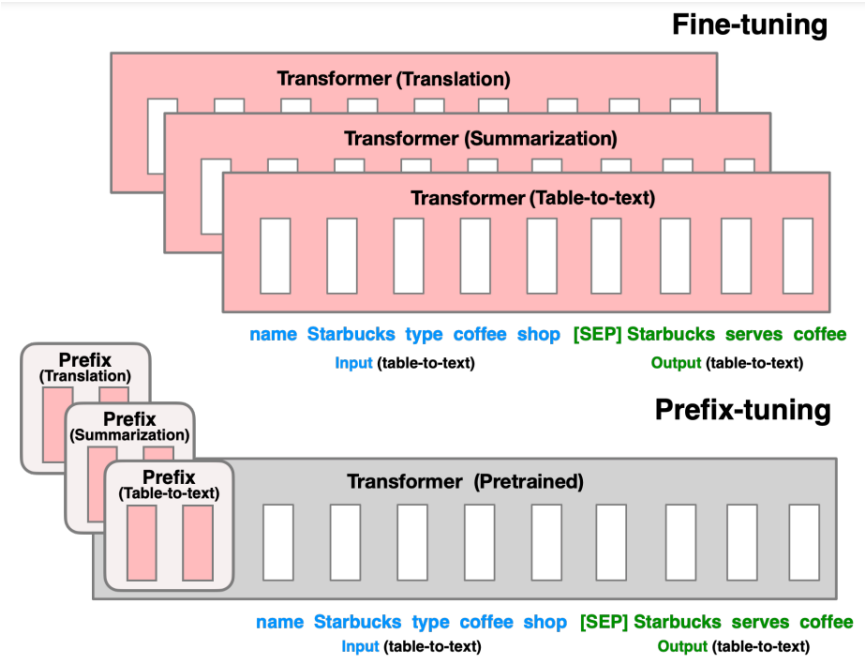
- **추론 시 대기 시간 (Inference Latency):**

  Adapter layers를 더함에 따라 추가적인 연산이 필요, 보통 하드웨어 병렬화로 latency를 낮추지만 adapter layer는 순차적으로 처리되어야 하기 때문에 (multihead attention값을 전달받아야 함)latency를 높인다.

	Batch Size Sequence Length |Θ|	32 512 0.5M	16 256 11M	1 128 11M
	Fine-Tune/LoRA	1449.4±0.8	338.0±0.6	19.8±2.7
(baseline)	$Adapter^L$	1482.0±1.0 (+2.2%)	354.8±0.5 (+5.0%)	23.9±2.1 (+20.7%)
	$Adapter^H$	1492.2±1.0 (+3.0%)	366.3±0.5 (+8.4%)	25.8±2.2 (+30.3%)

- **Optimizing some forms of the input layer activations**

  [prefix-tuning]



한계:

- **프롬프트 최적화의 어려움 (Directly Optimizing th Prompt is Hard)**

Prefix-tuning의 경우 파라미터가 명확히 수렴하지 않고 진동함(changes non-monotically)

adaptation을 위해 시퀀스 일부를 미리 떼어놔야 하기 때문에 다운스트림 태스크를 처리할 때 사용할 수 있는 시퀀스 길이를 줄인다.

Our Method

Low-Rank-Parameterized Update Matrices

가설: 가중치 업데이트는 낮은 intrinsic rank를 갖는다. (실제 실험 결과에 근거)

구조: down projection by A와 up projection by B를 곱해 만든 rank decomposition matrix로 ΔW를 근사한다. Full fine-tuning과의 차이점은 **업데이트할 가중치 행렬을 full-rank로 구성할 필요가 없다**고 가정하여 파라미터 수를 줄일 수 있게 된 것이다.

Low-rank r << d_model

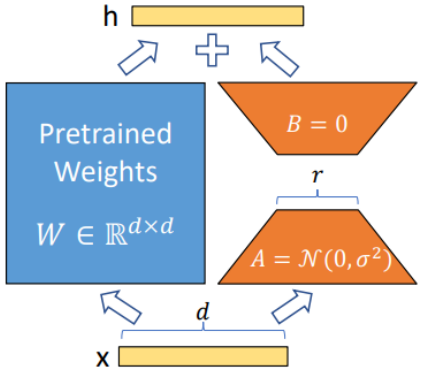


Figure 1: Our reparametrization. We only train A and B.

Forword pass:

$$h = W_0 x + \Delta W x = W_0 x + \overset{d\times r}{B}\overset{r\times k}{A} x$$

low-rank(r) decomposition

Detail:

학습 가능한 파라미터인 A, B 중 A에 대해서는 랜덤 가우시안 분포 값으로, B는 0으로 초기화 했다. 이후 α/r로 스케일링했다.

No Additional Inference Latency:

W_0과 BA의 차원은 같다. 따라서 다른 다운스트림 태스크에 대해 모델을 재조정하고 싶다면 기존 $W = W_0 + BA$에서 BA를 빼주고, 새로운 태스크에 대한 decomposition matrix B'A'를 더해주기만 하면 된다. (앞선 Adapter의 시퀀셜한 연산과 비교하면 차이가 두드러진다.)

Applying LoRA to Transformer

트랜스포머 아키텍쳐에는 self-attenstion module에 있는 4개의 가중치 행렬(W_q, W_k, W_v, W_o)과 두 개의 MLP 모듈으로 구성되는데, 이 연구에서는 단순성과 연산 효율성을 위해 오직 attention 가중치 행렬(W_q, W_k, W_v, W_o)에만 adaptation을 수행했다.

ㄹ

Conclusion and Future Work

LoRA는 1) 효율적으로 downstream task에 대해 전이 학습을 수행할 수 있게 해주며, 2) 추론 시에 대기 시간을 발생시키지 않고 3) 태스크 전환에도 용이하다.

추후 1) 다른 adaptation 방법론과 결합하거나, 2) 사전 학습 결과로 얻어진 특징들이 downstream task에 전이되는 구체적인 과정과 원리에 대한 탐구가 필요하다. 3) 또한 LoRA 행렬을 적용할 가중치 선정에 대한 원리를 구축하거나 4) 사전 학습 파라미터인 W 또한 deficient rank로 구성하고도 성능을 유지할 수 있는지 탐구해볼 수 있겠다.