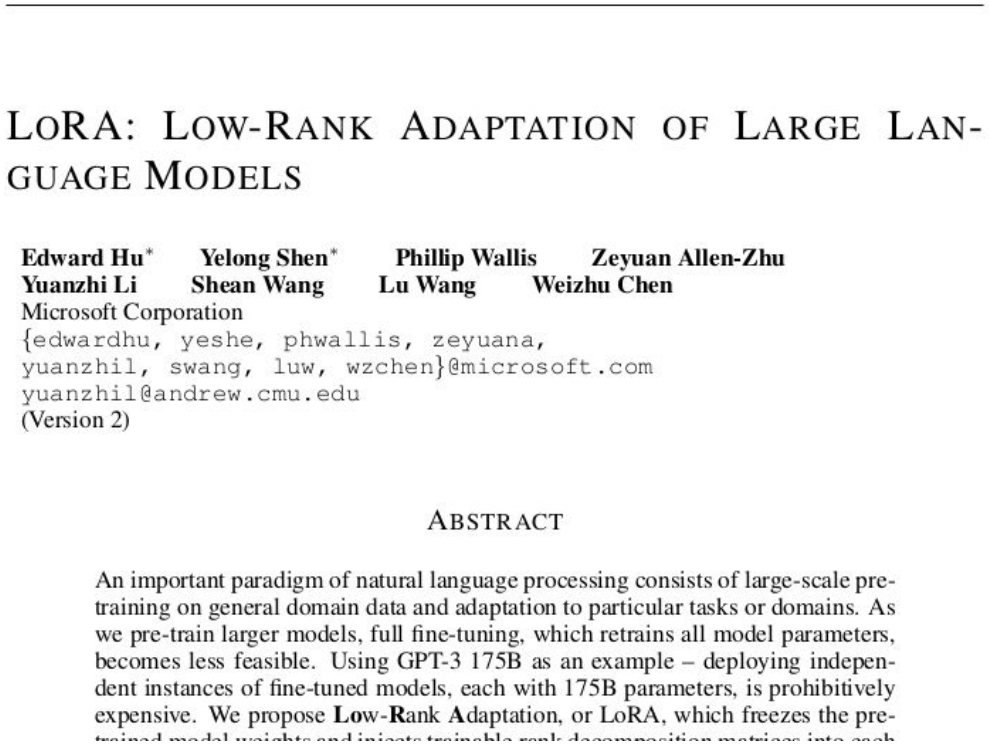
LoRA 기법 리뷰
(Low-Rank Adaptation of Large Language Models)

에이아이스쿨(AISchool) 대표
양진호 (솔라리스)
http://aischool.ai
http://solarisailab.com

LoRA Paper

- Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." arXiv preprint arXiv:2106.09685 (2021).
- https://arxiv.org/abs/2106.09685



LoRA: Low-Rank Adaptation of Large Language Models

Edward Hu* **Yelong Shen*** **Phillip Wallis** **Zeyuan Allen-Zhu**
Yuanzhi Li **Shean Wang** **Lu Wang** **Weizhu Chen**
Microsoft Corporation
{edwardhu, yeshe, phwallis, zeyuana,
yuanzhil, swang, luw, wzchen}@microsoft.com
yuanzhil@andrew.cmu.edu
(Version 2)

] 16 Oct 2021

Abstract

An important paradigm of natural language processing consists of large-scale pre-training on general domain data and adaptation to particular tasks or domains. As we pre-train larger models, full fine-tuning, which retrains all model parameters, becomes less feasible. Using GPT-3 175B as an example – deploying independent instances of fine-tuned models, each with 175B parameters, is prohibitively expensive. We propose **Lo**w-**R**ank **A**daptation, or LoRA, which freezes the pre-

Overview

- LoRA의 핵심 idea : fine-tuning 과정시 pre-train이 끝난 파라미터 w_0를 고정하고 low-rank decomposition 세팅의 새로운 파라미터를 학습시킴

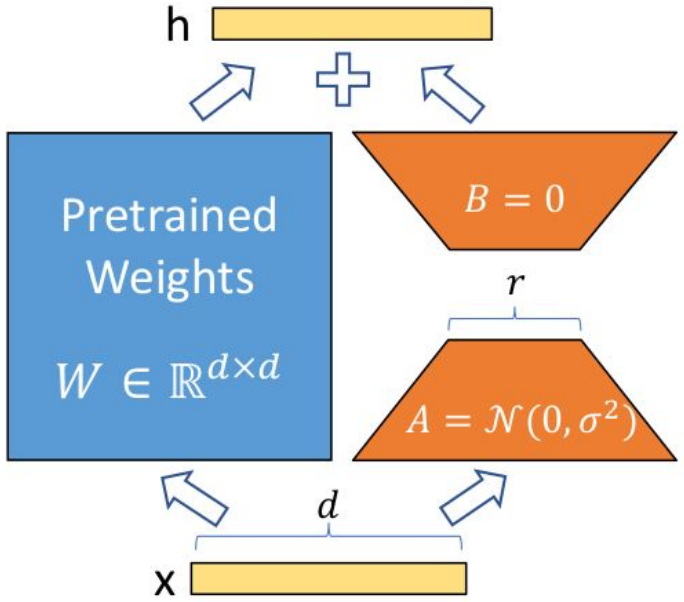


Figure 1: Our reparametrization. We only train A and B.

$$h = W_0 x + \Delta W x = W_0 x + BAx \qquad (3)$$

LoRA Paper

- LoRA Paper를 같이 살펴보면서 LoRA의 디테일한 내용들을 살펴봅시다!

Abstract

- 자연어 처리의 중요한 패러다임은 일반 도메인 데이터에서 대규모로 사전 훈련을 진행하고 특정 작업이나 도메인에 적응시키는 것입니다.
- 우리가 더 큰 모델을 사전 훈련(pre-train)할수록, 모든 모델 파라미터를 재훈련하는 전체 파인 튜닝은 더욱 실행하기 힘들어집니다. 175B의 GPT-3을 예로 들면 – 독립적으로 파인 튜닝된 모델의 인스턴스를 배포하는 것은 175B의 파라미터를 가질 때는 비용이 많이 듭니다.
- 우리는 Low-Rank Adaptation, 즉 LoRA를 제안합니다. 이는 사전 훈련된 모델 가중치를 고정(freezes)하고 각 변환 아키텍처 계층에 훈련 가능한 순위 분해 행렬(rank decomposition matrices)을 주입함으로써, 다운스트림 작업에 대한 훈련 가능한 파라미터의 수를 크게 줄입니다.

Abstract

- Adam으로 파인 튜닝된 GPT-3 175B와 비교할 때, LoRA는 훈련 가능한 파라미터 수를 10,000배, GPU 메모리 요구 사항을 3배 줄일 수 있습니다.
- LoRA는 RoBERTa, DeBERTa, GPT-2 및 GPT-3에서 파인 튜닝과 동등하거나 그 이상의 성능을 보이며, 훈련 가능한 파라미터가 더 적고, 높은 훈련 처리량을 가지며, *어댑터와 달리 추가적인 추론 지연*이 없습니다.
- 또한, 언어 모델 적응에서의 순위 결핍(rank-deficiency)에 관한 경험적 조사도 제공하여 LoRA의 효과에 대한 인사이트를 제공합니다.
- 우리는 PyTorch 모델과 LoRA의 통합을 용이하게 하는 패키지를 공개하며, RoBERTa, DeBERTa, GPT-2의 구현 및 모델 체크포인트를 https://github.com/microsoft/LoRA 에서 제공합니다.

1. Introduction

- 자연어 처리의 많은 응용 분야는 대규모의 *하나의(one)* 사전 훈련된 언어 모델을 *여러(multiple)* 하위 응용 프로그램에 적응시키는 데 의존하고 있습니다. 이러한 적응은 주로 *파인 튜닝(fine-tuning)*을 통해 이루어지며, 이는 사전 훈련(pre-trained)된 모델의 모든 파라미터를 업데이트합니다.

- 파인 튜닝의 주요 단점은 새로운 모델이 원래 모델만큼의 많은 파라미터를 포함한다는 것입니다. 몇 달마다 더 큰 모델이 훈련됨에 따라, 이는 GPT-2 (Radford 등, b)나 RoBERTa large (Liu 등, 2019)에 대한 단순한 "불편"에서 1750억 개의 훈련 가능한 파라미터를 가진 GPT-3 (Brown 등, 2020)에 대한 중대한 배포 도전 과제로 변하게 됩니다.

- 많은 사람들은 일부 파라미터만 적응시키거나 새로운 작업을 위한 외부 모듈을 학습함으로써 이를 완화하려고 했습니다. 이 방식을 통해, 우리는 각 작업에 대한 사전 훈련된 모델에 추가로 작업 특정 파라미터를 작은 수만 저장하고 불러올 필요가 있어, 배포 시 운영 효율성을 크게 향상시킵니다.

- 그러나 기존 기술들은 종종 모델의 깊이를 확장함으로써 추론 지연을 도입하거나 (Houlsby 등, 2019; Rebuffi 등, 2017), 모델의 사용 가능한 시퀀스 길이를 줄입니다 (Li & Liang, 2021; Lester 등, 2021; Hambardzumyan 등, 2020; Liu 등, 2021) (섹션 3).

- 더 중요한 것은, 이러한 방법들이 종종 파인 튜닝 기준에 맞지 않아 효율성과 모델 품질 사이에 타협을 강요하게 됩니다.

1. Introduction

- 우리는 Li 등 (2018a); Aghajanyan 등 (2020)의 연구에서 영감을 얻었습니다. 이 연구들은 학습된 과도하게 매개변수화된(over-parametrized) 모델이 실제로는 낮은 본질적 차원(low intrinsic dimension)에서 존재함을 보여줍니다.
- 우리는 모델 적응 동안의 가중치 변화 또한 낮은 "본질적 순위"(intrinsic rank)를 가진다고 가설을 세웠고, 이로 인해 제안된 Low-Rank Adaptation (LoRA) 접근법을 도출하게 되었습니다.
- LoRA는 그림 1에서 보여지는 것처럼 사전 훈련된 가중치를 고정(frozen)시킨 상태에서, 적응 동안의 밀집 계층의 변화의 순위 분해 행렬(rank decomposition matrices)을 최적화함으로써 신경망의 일부 밀집 계층을 간접적으로 훈련시키는 것을 허용합니다.
- GPT-3 175B를 예로 들면, 전체 순위 (즉, d)가 12,288이나 되는 높은 수치에서도 매우 낮은 순위(low rank) (즉, 그림 1의 r이 1 또는 2가 될 수 있음)만으로도 충분하다는 것을 보여줍니다.
- 이로 인해 LoRA는 저장 및 연산 효율성 모두를 갖추게 됩니다.

1. Introduction

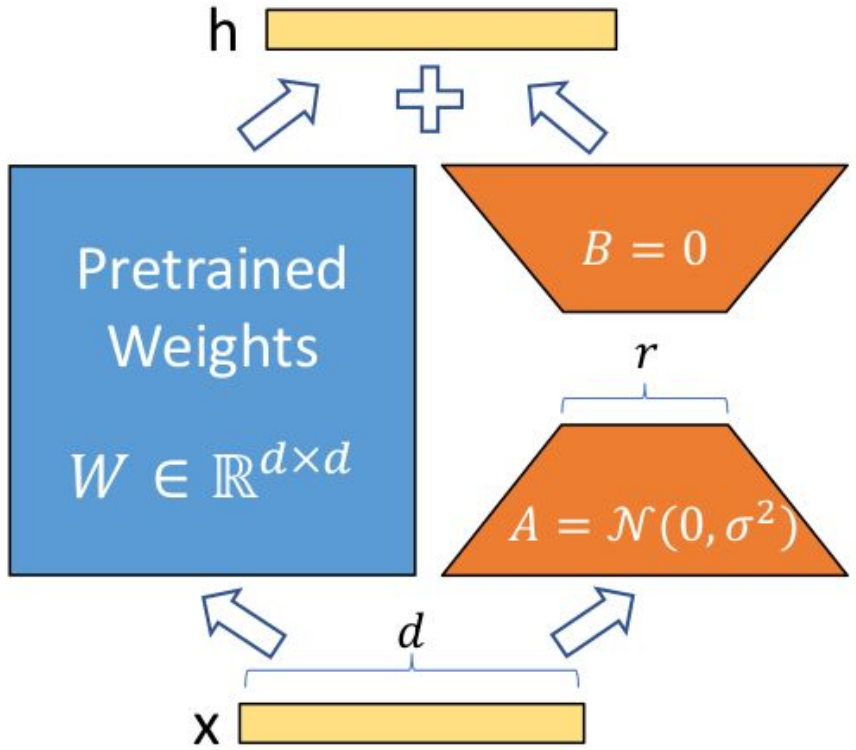


Figure 1: Our reparametrization. We only train A and B.

1. Introduction

• LoRA는 여러 주요한 장점을 갖고 있습니다.

1. 사전 훈련된 모델은 공유되어 여러 다른 작업에 대한 많은 작은 LoRA 모듈을 구축하는 데 사용될 수 있습니다. 공유 모델을 고정(freeze)시키고 그림 1의 행렬 A와 B를 교체함으로써 효율적으로 작업을 전환할 수 있습니다. 이로 인해 저장 요구 사항과 작업 전환 오버헤드(task-switching overhead)가 크게 감소합니다.
2. LoRA는 훈련을 더 효율적으로 만들며, 대부분의 파라미터에 대한 그래디언트를 계산하거나 최적화 상태를 유지할 필요가 없기 때문에 적응형 최적화기를 사용할 때 하드웨어 진입 장벽을 최대 3배까지 낮춥니다. 대신, 우리는 주입된 훨씬 작은 저랭크 행렬(low-rank matrices)만 최적화합니다.
3. 우리의 간단한 선형 설계는 배포할 때 훈련 가능한 행렬을 고정된(frozen) 가중치와 합치게 해서 완전히 파인 튜닝된 모델에 비해 *추론 지연을 도입하지 않게* 합니다.
4. LoRA는 많은 이전 방법과 독립적이며(orthogonal), 그 중 많은 것과 결합될 수 있습니다. 예를 들어, prefix-tuning과 같은 방법과 함께 사용될 수 있습니다. 부록 E에서 예시를 제공합니다.

1. Introduction

- 용어 및 관례(Terminologies and Conventions) : 우리는 Transformer 아키텍처를 자주 언급하며 그 차원에 대한 전통적인 용어를 사용합니다. Transformer 계층의 입력 및 출력 차원 크기를 d_model 로 부릅니다.
- 자기 주의 모듈(self-attention module)에서의 쿼리/키/값/출력 투영 행렬을 참조하기 위해 W_q , W_k , W_v , 및 W_o 를 사용합니다.
- W 또는 W_0 는 사전 훈련된 가중치 행렬을 나타내고 ΔW는 적응 중에 누적된 그래디언트 업데이트를 나타냅니다.
- LoRA 모듈의 순위(rank)를 나타내기 위해 r을 사용합니다. (Vaswani 등, 2017; Brown 등, 2020)에서 제시한 관례를 따르며 모델 최적화를 위해 Adam (Loshchilov & Hutter, 2019; Kingma & Ba, 2017)을 사용하고, Transformer MLP 전방 피드 포워드 차원은 d_ffn = 4 × d_model 로 사용합니다.

2. PROBLEM STATEMENT

- 우리의 제안은 훈련 목표에 중립적(agnostic)이지만, 우리의 주요 동기는 언어 모델링에 있습니다. 아래는 언어 모델링 문제와 특히 작업 특정 프롬프트가 주어진 조건부 확률의 최대화에 대한 간략한 설명입니다.
- Φ로 매개변수화된 사전 훈련된 자동 회귀 언어 모델 P_Φ(y|x)가 주어진다고 가정해봅시다.
- 예를 들면, P_Φ(y|x)는 Transformer 아키텍처 (Vaswani 등, 2017)를 기반으로 하는 GPT (Radford 등, b; Brown 등, 2020)와 같은 일반적인 다중 작업 학습자(multi-task learner)일 수 있습니다.
- 이런 사전 훈련된 모델을 다양한 조건부 텍스트 생성 작업, 예를 들면 요약, 기계 읽기 이해 (machine reading comprehension) (MRC), 자연어에서 SQL(natural language to SQL)(NL2SQL)로의 변환 등에 적용하는 것을 고려해봅시다.
- 각 하류 작업(downstream task)은 컨텍스트-대상 쌍의 훈련 데이터셋으로 표현됩니다: Z = {(x_i, y_i)}_i=1,..,N , 여기서 x_i 와 y_i 모두 토큰의 시퀀스입니다.
- 예를 들면, SQL(NL2SQL)에서 x_i 는 자연어 쿼리이고 y_i 는 그에 상응하는 SQL 명령어입니다; 요약의 경우, x_i 는 기사의 내용이고 y_i 는 그 기사의 요약입니다.

2. PROBLEM STATEMENT

- 완전한 파인 튜닝 동안, 모델은 사전 훈련된 가중치 Φ_0 로 초기화되며 조건부 언어 모델링 목표를 최대화하기 위해 그래디언트를 반복적으로 따라 Φ_0 + ΔΦ로 업데이트됩니다:

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

- 완전한 파인 튜닝(full fine-tuning)의 주요 단점 중 하나는 각 하류 작업마다 |ΔΦ|의 차원이 |Φ_0|와 같은 다른 파라미터 집합 ΔΦ를 학습한다는 것입니다.
- 따라서 사전 훈련된 모델이 크다면 (예: |Φ_0 | ≈ 1750억인 GPT-3와 같이), 파인 튜닝된 모델의 많은 독립적인 인스턴스를 저장하고 배포하는 것이 어려울 수 있으며, 실현 가능한지 여부도 불확실합니다.

2. PROBLEM STATEMENT

- 이 논문에서는, 작업 특정 파라미터 증가량 ΔΦ = ΔΦ(Θ)가 훨씬 더 작은 크기의 파라미터 집합 Θ에 의해 추가로 인코딩되는 더 파라미터 효율적인 접근법 (parameter-efficient approach)을 채택합니다. 여기서 |Θ|는 |Φ_0 |보다 훨씬 작습니다. |Θ| << |Φ_0 |
- 따라서 ΔΦ를 찾는 작업은 Θ에 대한 최적화가 됩니다:

$$\max_{\Theta} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(p_{\Phi_0+\Delta\Phi(\Theta)}(y_t|x,y_{<t})\right) \tag{2}$$

- 이후의 섹션에서는, 계산 및 메모리 효율적인 저랭크 표현(low-rank representation)을 사용하여 ΔΦ를 인코딩하는 방법을 제안합니다.
- 사전 훈련된 모델이 GPT-3 175B일 때, 훈련 가능한 파라미터의 수 |Θ|는 |Φ_0 |의 0.01%만큼 작을 수 있습니다.

3. AREN'T EXISTING SOLUTIONS GOOD ENOUGH?

- 우리가 다루려는 문제는 결코 새로운 것이 아닙니다. 전이 학습이 시작된 이후 수십 개의 연구가 모델 적응(model-adaption)을 더 파라미터(parameter) 및 계산 효율적(compute-efficient)으로 만들기 위해 노력해왔습니다.
- 잘 알려진 연구들 중 일부에 대한 조사는 6장에서 확인할 수 있습니다.
- 언어 모델링을 예로 들면, 효율적인 적응(efficient adaptations)에 있어 두 가지 두드러진 전략이 있습니다: 어댑터 레이어 추가 (Houlsby 등, 2019; Rebuffi 등, 2017; Pfeiffer 등, 2021; Rücklé 등, 2020) 또는 입력 레이어 활성화의 어떤 형태를 최적화하는 것 (Li & Liang, 2021; Lester 등, 2021; Hambardzumyan 등, 2020; Liu 등, 2021).
- 그러나 두 전략 모두, 특히 대규모(large-scale) 및 지연 시간(latency-sensitive)에 민감한 프로덕션 상황에서 제한 사항을 가지고 있습니다.

3. AREN'T EXISTING SOLUTIONS GOOD ENOUGH?

- 어댑터 레이어는 추론 지연을 초래합니다(Adapter Layers Introduce Inference Latency). 어댑터에는 여러 변형이 있습니다. 우리는 Transformer 블록당 두 개의 어댑터 레이어를 갖는 Houlsby 등(2019)의 오리지날 디자인과 추가적인 LayerNorm(Ba 등, 2016)을 포함해 블록당 하나만 가진 Lin 등(2020)의 최근 디자인에 주목합니다.

- 레이어를 가지치기(pruning layers)하거나 다중 작업 설정을 활용하여 전체 지연 시간을 줄일 수 있지만(Rücklé 등, 2020; Pfeiffer 등, 2021), 어댑터 레이어의 추가 계산을 직접 우회하는 방법은 없습니다.

- 어댑터 레이어는 작은 병목 차원을 갖도록 설계되어 매개 변수가 적다(때로는 원래 모델의 <1%)는 것처럼 보이지만, 이로 인해 추가할 수 있는 FLOPs가 제한됩니다.

- 그러나 큰 신경망은 하드웨어 병렬성에 의존하여 지연 시간을 낮게 유지하며, 어댑터 레이어는 순차적으로 처리되어야 합니다. 이것은 일반적으로 배치 크기가 1 정도로 작은 온라인 추론 설정에서 차이를 만듭니다.

- 모델 병렬성이 없는 일반적인 시나리오에서, 예를 들어 GPT-2(Radford 등, b) medium 모델을 단일 GPU에서 추론 실행시, 아주 작은 병목 차원을 사용할 때도 어댑터를 사용하면 지연 시간이 눈에 띄게 증가하는 것을 확인할 수 있습니다(표 1 참조).

3. AREN'T EXISTING SOLUTIONS GOOD ENOUGH?

- 이 문제는 Shoeybi 등(2020), Lepikhin 등(2020)에서 수행한 것처럼 모델을 샤딩할 필요가 있을 때 더욱 악화됩니다.
- 왜냐하면 추가적인 깊이는 어댑터 매개 변수를 중복으로 많이 저장하지 않는 한 AllReduce와 Broadcast와 같은 동기(synchronous) GPU 작업을 더 많이 필요로 하기 때문입니다.

3. AREN'T EXISTING SOLUTIONS GOOD ENOUGH?

Batch Size Sequence Length $\|\Theta\|$	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
$\text{Adapter}^{\text{L}}$	1482.0±1.0 (+2.2%)	354.8±0.5 (+5.0%)	23.9±2.1 (+20.7%)
$\text{Adapter}^{\text{H}}$	1492.2±1.0 (+3.0%)	366.3±0.5 (+8.4%)	25.8±2.2 (+30.3%)

Table 1: Infernece latency of a single forward pass in GPT-2 medium measured in milliseconds, averaged over 100 trials. We use an NVIDIA Quadro RTX8000. "$|\Theta|$" denotes the number of trainable parameters in adapter layers. $\text{Adapter}^{\text{L}}$ and $\text{Adapter}^{\text{H}}$ are two variants of adapter tuning, which we describe in Section 5.1. The inference latency introduced by adapter layers can be significant in an online, short-sequence-length scenario. See the full study in Appendix B.

AI School.

3. AREN'T EXISTING SOLUTIONS GOOD ENOUGH?

- 프롬프트를 직접 최적화하는 것은 어렵다(Directly Optimizing the Prompt is Hard). prefix tuning (Li & Liang, 2021)을 예로 든 다른 방향은 다른 도전을 직면하고 있다.
- 우리는 prefix tuning이 최적화하기 어렵다는 것과 그 성능이 훈련 가능한 매개변수에 따라 비모노토닉(non-monotonically)하게 변한다는 것을 관찰했다.
- 이는 원래의 논문에서도 유사하게 관찰된 내용을 확인한다. 더 근본적으로, 적응을 위해 시퀀스 길이의 일부를 예약하게 되면, downstream 작업을 처리하기 위해 사용 가능한 시퀀스 길이가 줄어들게 된다. 이것이 프롬프트 튜닝이 다른 방법에 비해 덜 효과적이게 만든다고 우리는 추측한다.
- 작업 성능에 대한 연구는 섹션 5에서 다룬다.

4. OUR METHOD

- 우리는 LoRA의 간단한 설계와 그 실용적인 이점을 설명합니다.
- 여기서 제시된 원칙은 딥러닝 모델의 모든 밀집 레이어(dense layer)에 적용될 수 있지만, 실험에서는 Transformer 언어 모델의 특정 가중치(weights)에만 초점을 맞추어 모티베이션을 제공하는 사례로 집중합니다.

4.1 LOW-RANK-PARAMETRIZED UPDATE MATRICES

- 신경망에는 행렬 곱셈을 수행하는 많은 밀집 레이어(dense layer)가 포함되어 있습니다.
- 이러한 레이어의 가중치 행렬은 대체로 full-rank를 가지고 있습니다. 특정 작업에 적응할 때, Aghajanyan 등(2020)은 사전 훈련된 언어 모델이 낮은 "본질적 차원"(intrinsic dimension)을 가지고 있으며 작은 부분 공간으로의 무작위 투영에도 불구하고 효율적으로 학습할 수 있다고 보여줍니다.
- 이를 바탕으로, 우리는 적응 도중 가중치의 업데이트도 낮은 "본질적 순위"(intrinsic rank)를 가질 것이라고 가정합니다. 사전 훈련된 가중치 행렬 W_0 ∈ R^d×k 에 대해, 우리는 그 업데이트를 낮은 순위 분해 (low-rank decomposition)를 사용하여 표현함으로써 제한하고자 합니다.
- W_0 + ΔW = W_0 + BA, 여기서 B ∈ R d×r , A ∈ R r×k 이며, 순위(rank) r << min(d, k)입니다.
- 훈련 중에는 W_0 가 고정되어 경사 업데이트를 받지 않으며, A와 B는 훈련 가능한 매개변수를 포함합니다.
- W_0 와 ΔW = BA는 동일한 입력으로 곱해지며, 각각의 출력 벡터는 좌표별로 합산됩니다. h = W_0x의 경우, 우리의 수정된 전방 패스는 다음과 같습니다:

$$h = W_0x + \Delta W x = W_0x + BAx \tag{3}$$

AI School.

4.1 LOW-RANK-PARAMETRIZED UPDATE MATRICES

- 우리는 그림 1에서 재매개변수화를 설명합니다.
- A에는 무작위 가우시안 초기화를 사용하고 B에는 0을 사용하여 훈련 시작 시 $\Delta W = BA$는 0입니다.
- 그런 다음 α/r로 $\Delta W x$를 스케일링합니다. 여기서 α는 r의 상수입니다.
- Adam으로 최적화할 때, 초기화를 적절히 스케일링한다면 α를 조정하는 것은 학습률을 조정하는 것과 대략적으로 같습니다.
- 결과적으로, 우리는 α를 첫 번째로 시도하는 r에 설정하고 조정하지 않습니다. 이러한 스케일링은 r을 변경할 때 하이퍼파라미터를 재조정할 필요를 줄여줍니다 (Yang & Hu, 2021).

4.1 LOW-RANK-PARAMETRIZED UPDATE MATRICES

- 전체 미세조정의 일반화(A Generalization of Full Fine-tuning). 미세 조정 (fine-tuning)의 더 일반적인 형태는 사전 훈련된 파라미터의 부분 집합을 훈련하는 것을 허용합니다.
- LoRA는 한 걸음 더 나아가서 적응 도중 가중치 행렬에 대한 누적된 기울기 업데이트가 전체 랭크를 갖도록 요구하지 않습니다.
- 이는 LoRA를 모든 가중치 행렬에 적용하고 모든 바이어스를 훈련시킬 때, LoRA의 랭크 r을 사전 훈련된 가중치 행렬의 랭크로 설정함으로써 전체 미세 조정의 표현력을 대략적으로 회복한다는 것을 의미합니다.
- 다시 말해, 훈련 가능한 파라미터의 수를 늘릴수록, LoRA 훈련은 대략적으로 원래 모델의 훈련으로 수렴하며, 어댑터 기반 방법은 MLP로 수렴하고, 접두사 기반 방법은 긴 입력 시퀀스를 처리할 수 없는 모델로 수렴합니다.

4.1 LOW-RANK-PARAMETRIZED UPDATE MATRICES

- 추가적인 추론 지연 없음(No Additional Inference Latency). 우리가 프로덕션 배포할 때, 우리는 명시적으로 W=W_0+BA를 계산하고 저장하고 평소와 같이 추론을 수행할 수 있습니다.
- 주목할 것은 W_0와 BA 모두 R^d×k 차원이라는 것입니다.
- 우리가 다른 하위 작업으로 전환할 필요가 있을 때, 우리는 BA를 빼서 W_0를 복구하고 다른 B′A′를 더할 수 있습니다, 이는 매우 적은 메모리 오버헤드로 빠른 작업입니다.
- 중요한 것은, 이것이 구성에 의해 미세 조정된 모델과 비교하여 추론 중에 추가적인 지연을 도입하지 않는다는 것을 보장한다는 것입니다.

4.2 APPLYING LORA TO TRANSFORMER

- 원칙적으로, 우리는 신경망에서 훈련 가능한 매개변수의 수를 줄이기 위해 어떤 가중치 행렬의 부분집합에도 LoRA를 적용할 수 있습니다.
- Transformer 아키텍처에서는 self-attention 모듈에 네 개의 가중치 행렬(W_q, W_k, W_v, W_o)과 MLP 모듈에 두 개가 있습니다.
- 출력 차원이 보통 attention head로 나뉘어져 있더라도, 우리는 W_q (또는 W_k, W_v)를 d_model×d_model의 차원을 가진 단일 행렬로 취급합니다.
- 우리는 단순성과 매개변수 효율성을 위해 하위 작업에 대한 주의 가중치(attention weights)만 적용하는 것으로 제한하고 MLP 모듈을 고정합니다(즉, 하위 작업에서 훈련되지 않습니다).
- 우리는 또한 7.1 절에서 Transformer 내에서 다양한 타입의 attention 가중치 행렬을 적용하는 효과에 대해 더 자세히 연구합니다. MLP 레이어, LayerNorm 레이어, 그리고 바이어스를 적용하는 실증적인 조사는 향후 연구로 남겨둡니다.

4.2 APPLYING LORA TO TRANSFORMER

- 실용적인 이점과 한계(Practical Benefits and Limitations.). 가장 큰 이점은 메모리와 저장 공간 사용의 감소에서 나옵니다.
- Adam으로 훈련된 큰 Transformer의 경우, 고정된 매개변수에 대한 최적화 상태를 저장할 필요가 없기 때문에 r << d_model 일 때 VRAM 사용량을 최대 2/3까지 줄일 수 있습니다.
- GPT-3 175B에서는 훈련 중의 VRAM 소비를 1.2TB에서 350GB로 줄입니다.
- r=4이고 쿼리와 값 투영 행렬만 적응하는 경우, 체크포인트 크기는 대략 10,000배 (350GB에서 35MB) 감소합니다.
- 이로 인해 훨씬 적은 GPU로 훈련할 수 있으며 I/O 병목 현상을 피할 수 있습니다.
- 또 다른 이점은 모든 매개변수 대신 LoRA 가중치만 교환하여 배포 중에 작업 간 전환이 훨씬 저렴하다는 것입니다.
- 이를 통해 사전 훈련된 가중치를 VRAM에 저장하는 기계에서 즉석에서 교체할 수 있는 많은 맞춤 모델을 생성할 수 있습니다. 또한, 대다수의 매개변수에 대한 그라디언트를 계산할 필요가 없기 때문에 GPT-3 175B에서의 훈련 중 25%의 속도 향상을 관찰했습니다.

8. CONCLUSION AND FUTURE WORK

- 거대한 언어 모델을 미세 조정(Fine-tuning)하는 것은 필요한 하드웨어와 다양한 작업에 대한 독립적인 인스턴스를 호스팅하는 엄청난 저장/전환 비용이 필요합니다.
- 우리는 LoRA를 제안합니다. 이는 추론 지연(inference latency)을 도입하지 않고 입력 시퀀스 길이를 줄이지 않으면서도 높은 모델 품질을 유지하는 효율적인 적응 전략입니다.
- 중요한 것은, 대다수의 모델 매개변수를 공유함으로써 서비스로 배포될 때 빠른 작업 전환이 가능하게 합니다.
- 우리는 Transformer 언어 모델에 중점을 두었지만, 제안된 원칙들은 밀집된 레이어가 있는 어떤 신경망에도 일반적으로 적용 가능합니다.

8. CONCLUSION AND FUTURE WORK

- 미래의 연구 방향은 많습니다.

1. LoRA는 다른 효율적인 적응 방법과 결합될 수 있으며, 이는 독립적인 개선(orthogonal improvement)을 제공할 수 있습니다.
2. 미세 조정 또는 LoRA 뒤의 메커니즘은 분명하지 않습니다 – 사전 훈련 중에 학습된 특징들이 하위 작업에서 어떻게 잘 수행되도록 변환되는가? 전체 미세 조정보다 LoRA를 사용하면 이 문제를 더 쉽게 답변할 수 있다고 믿습니다.
3. 우리는 주로 LoRA를 적용할 가중치 행렬을 선택하기 위해 휴리스틱에 의존합니다. 이것을 할 더 원칙적인 방법이 있을까요?
4. 마지막으로, ΔW의 순위 결함(rank-deficiency)은 W도 순위 결함일 수 있음을 나타냅니다, 이것 또한 미래의 연구를 위한 영감의 원천이 될 수 있습니다.