Prefix-Tuning 기법 리뷰
(Prefix-Tuning: Optimizing Continuous Prompts for Generation)

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

Prefix-tuning Paper

- Li, Xiang Lisa, and Percy Liang. "Prefix-tuning: Optimizing continuous prompts for generation." arXiv preprint arXiv:2101.00190 (2021).
- https://arxiv.org/abs/2101.00190

Prefix-Tuning: Optimizing Continuous Prompts for Generation

Xiang Lisa Li
Stanford University
xlisali@stanford.edu

Percy Liang
Stanford University
pliang@cs.stanford.edu

Abstract

Fine-tuning is the de facto way to leverage large pretrained language models to perform downstream tasks. However, it modifies all the language model parameters and therefore necessitates storing a full copy for each task. In this paper, we propose prefix-tuning, a lightweight alternative to fine-tuning for natural language generation tasks, which keeps language model parameters frozen, but optimizes a small *continuous task-specific* vector (called the prefix). Prefix-tuning draws inspiration from prompting, allowing subsequent tokens to attend to this prefix as if it were "virtual tokens". We apply prefix-tuning to GPT-2 for table-to-text generation and to BART for summarization. We find that by learning only 0.1% of the parameters, prefix-tuning obtains comparable performance in the full data setting, outperforms fine-tuning in low-data settings, and extrapolates better to examples with topics unseen during training.

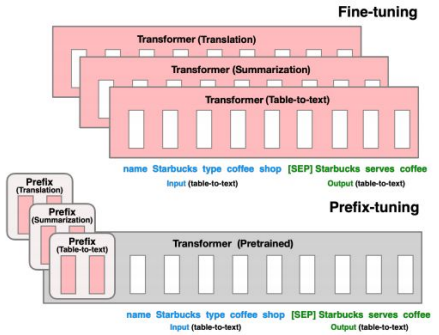

Figure 1: Fine-tuning (top) updates all Transformer parameters (the red Transformer box) and requires storing a full model copy for each task. We propose prefix-tuning (bottom), which freezes the Transformer parameters and only optimizes the prefix (the red prefix blocks). Consequently, we only need to store the prefix for each task, making prefix-tuning modular and space-efficient. Note that each vertical block denote transformer activations at one time step.

Overview

- Prefix-tuning의 핵심 idea : fine-tuning 과정시 pre-train이 끝난 파라미터 w_0를 고정하고 Prefix-tuning 세팅의 새로운 파라미터를 학습시킴

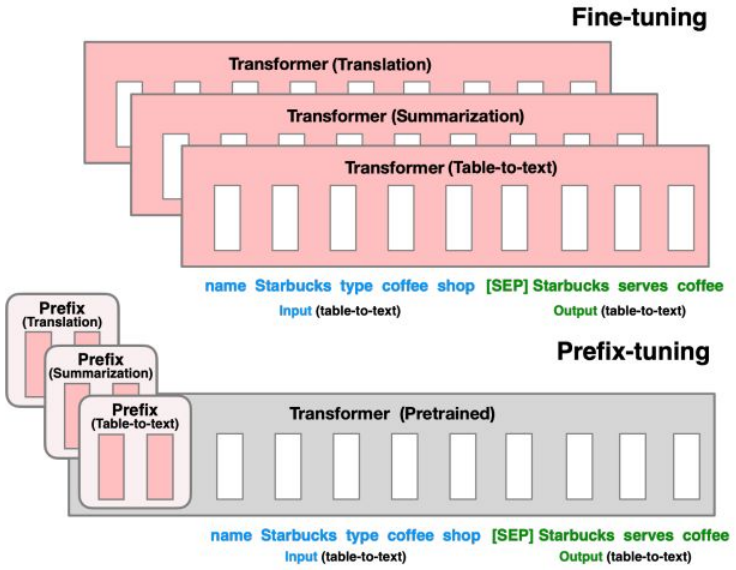

Figure 1: Fine-tuning (top) updates all Transformer parameters (the red Transformer box) and requires storing a full model copy for each task. We propose prefix-tuning (bottom), which freezes the Transformer parameters and only optimizes the prefix (the red prefix blocks). Consequently, we only need to store the prefix for each task, making prefix-tuning modular and space-efficient. Note that each vertical block denote transformer activations at one time step.

Prefix-tuning Paper

- Prefix-tuning Paper를 같이 살펴보면서 Prefix-tuning의 디테일한 내용들을 살펴봅시다!

Abstract

- 대부분의 사전 훈련된 언어 모델을 활용하여 하위 작업을 수행하기 위해 세부 조정(fine-tuning)이 주로 사용되고 있습니다.
- 그러나 이 방법은 언어 모델의 모든 매개변수를 수정하기 때문에 각 작업별로 전체 복사본을 저장해야 합니다.
- 본 논문에서는 자연어 생성 작업을 위한 세부 조정의 경량화된 대안으로 접두사 조정(prefix-tuning)을 제안합니다.
- 이 방법은 언어 모델의 매개변수를 고정시키고 작은 연속적인 작업별 벡터(continuous task-specific vector)(접두사(prefix)라고 함)만을 최적화합니다.

Abstract

- 접두사 조정(Prefix-tuning)은 프롬프팅에서 영감을 받아 이후 토큰이 이 접두사에 "가상 토큰"처럼 접근할 수 있게 합니다.
- 우리는 테이블-텍스트 생성(table-to-text generation)을 위해 GPT-2에 접두사 조정을 적용하고, 요약(summarization)을 위해 BART에 적용했습니다.
- 우리는 매개변수의 단 0.1%만 학습시켜도 접두사 조정(prefix-tuning)이 전체 데이터 설정에서 비슷한 성능을 얻고, 데이터가 적은 설정에서 세부 조정을 능가하며, 훈련 중에 보지 못한 주제의 예시에 더 잘 일반화된다는 것을 발견했습니다.

1. Introduction

- 세부 조정(fine-tuning)은 큰 규모의 사전 훈련된 언어 모델(LM) (Radford et al., 2019; Devlin et al., 2019)을 사용하여 하위 작업(예: 요약)을 수행하는 데 널리 사용되는 패러다임입니다. 그러나 이는 LM의 모든 매개변수를 업데이트하고 저장하는 것을 요구합니다.
- 따라서 큰 규모의 사전 훈련된 LMs에 의존하는 NLP 시스템을 구축하고 배포하려면 현재 각 작업에 대해 LM 매개변수의 수정된 복사본을 저장해야 합니다.
- 현재 LM의 큰 크기를 고려하면, 이것은 비용이 많이 드는 작업이 될 수 있습니다. 예를 들어, GPT-2는 774M의 매개변수를 가지고 있습니다 (Radford et al., 2019) 그리고 GPT-3는 175B의 매개변수를 가지고 있습니다 (Brown et al., 2020).

1. Introduction

- 이 문제에 대한 자연스러운 접근 방법은 경량화된 세부 조정(lightweight fine-tuning)입니다.
- 이 방법은 대부분의 사전 훈련된 매개변수를 고정시키고, 모델에 작은 훈련 가능한 모듈을 추가합니다.
- 예를 들어, 어댑터 조정(adapter-tuning) (Rebuffi et al., 2017; Houlsby et al., 2019)은 사전 훈련된 언어 모델의 레이어 사이에 추가적인 작업별 레이어를 삽입합니다. 어댑터 조정은 자연어 이해 및 생성 벤치마크에서 유망한 성능을 보이며, 단지 2-4%의 작업별 매개변수만 추가하면서도 세부 조정과 비슷한 성능을 달성합니다 (Houlsby et al., 2019; Lin et al., 2020).
- 극단적인 예로, GPT-3 (Brown et al., 2020)은 어떤 작업별 조정(task-specific tuning) 없이도 배포될 수 있습니다.
- 대신 사용자는 자연어 작업 지시(예: TL;DR은 요약을 위한 것)와 작업 입력에 대한 몇 가지 예시를 앞에 붙여; 그 후 LM에서 출력을 생성합니다. 이 접근법은 문맥 내 학습(in-context learning) 또는 프롬프팅(prompting)으로 알려져 있습니다.

1. Introduction

- 이 논문에서는 프롬프팅에서 영감을 받아 자연어 생성(NLG) 작업을 위한 세부 조정의 경량화된 대안인 접두사 조정(prefix-tuning)을 제안합니다.
- 그림 1에 나타난 데이터 테이블의 텍스트 설명을 생성하는 작업을 생각해보세요.
- 여기서 작업 입력은 선형화된 테이블(예: "name: Starbucks | type: coffee shop")이고 출력은 텍스트 설명(예: “Starbucks는 커피를 제공합니다.”)입니다.
- 접두사 조정(Prefix-Tuning)은 입력에 연속적인 작업별 벡터(continuous task-specific vectors)의 순서를 앞에 붙이며, 우리는 이를 그림 1(아래)의 빨간 블록으로 나타낸 접두사(prefix)라고 합니다.
- 이후 토큰들은 Transformer가 “가상 토큰”들의 순서처럼 접두사에 주의를 기울일 수 있지만, 프롬프팅과 달리 접두사는 전적으로 실제 토큰에 해당하지 않는 자유 매개변수(free parameters)로 구성됩니다.
- 그림 1(위)의 세부 조정과 달리, 모든 Transformer 매개변수를 업데이트하고 따라서 각 작업에 대해 조정된 모델의 복사본을 저장해야 하는 것에 비해, 접두사 조정(prefix-tuning)은 접두사만 최적화합니다.
- 결과적으로, 우리는 큰 Transformer의 하나의 복사본과 학습된 작업별 접두사만 저장해야 하므로, 각 추가 작업에 대한 오버헤드가 매우 작아집니다(예: 테이블-텍스트에 대해 250K 매개변수).

1. Introduction

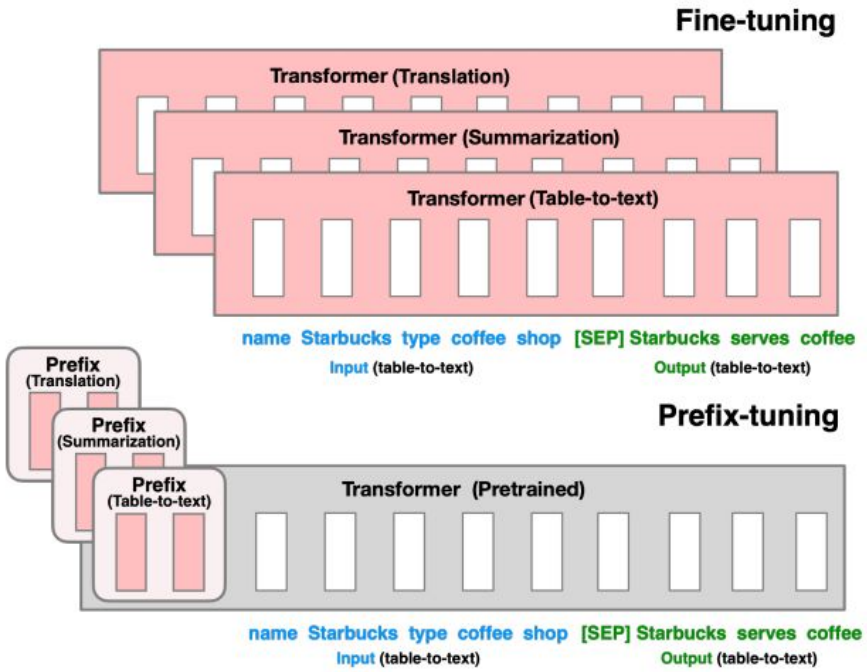

Figure 1: Fine-tuning (top) updates all Transformer parameters (the red Transformer box) and requires storing a full model copy for each task. We propose prefix-tuning (bottom), which freezes the Transformer parameters and only optimizes the prefix (the red prefix blocks). Consequently, we only need to store the prefix for each task, making prefix-tuning modular and space-efficient. Note that each vertical block denote transformer activations at one time step.

1. Introduction

- 세부 조정(fine-tuning)과는 대조적으로, 접두사 조정(prefix-tuning)은 모듈식입니다: 우리는 하류의 LM을 안내하는 상류 접두사를 훈련시키며, LM은 수정되지 않습니다.
- 따라서 하나의 LM은 한 번에 여러 작업을 지원할 수 있습니다. 개인화의 맥락에서 작업이 다른 사용자에게 해당할 때(Shokri와 Shmatikov, 2015; McMahan 등, 2016), 우리는 각 사용자의 데이터만을 훈련시킨 각 사용자를 위한 별도의 접두사를 가질 수 있으므로 데이터 간 교차 오염을 피할 수 있습니다.
- 더욱이, 접두사 기반의 구조는 하나의 배치에서 여러 사용자/작업의 예시를 처리하는 것도 가능하게 합니다, 이는 다른 경량화된 세부 조정 접근법으로는 가능하지 않습니다.

1. Introduction

- 우리는 GPT-2를 사용한 테이블-텍스트 생성(table-to-text generation)과 BART를 사용한 요약 생성(abstractive summarization)에 대해 접두사 조정(prefix-tuning)을 평가합니다.
- 저장 공간 측면에서, 접두사 조정은 세부 조정(fine-tuning)보다 1000배 적은 매개변수를 저장합니다.
- 전체 데이터셋에서 훈련될 때의 성능 측면에서는, 테이블-텍스트(§6.1)에 대해 접두사 조정과 세부 조정은 비슷한 성능을 보이지만, 요약(§6.2)의 경우 접두사 조정은 약간의 성능 저하가 있습니다.
- 데이터가 적은 설정에서는 접두사 조정이 평균적으로 두 작업 모두에서 세부 조정을 앞섭니다(§6.3).
- 또한, 접두사 조정은 보지않은 주제를 가진 테이블(테이블-텍스트를 위해) 및 기사(요약을 위해)에 대해 더 잘 추론합니다.(§6.4).

3. Problem Statement

- 조건부 생성 작업을 고려해보면, 입력은 맥락 x이며 출력 y는 토큰의 연속열입니다.
- 우리는 그림 2(오른쪽)에 표시된 두 가지 작업에 중점을 둡니다:
- 테이블-텍스트에서 x는 선형화된 데이터 테이블에 해당하고 y는 텍스트 설명입니다;
- 요약에서 x는 기사이며 y는 짧은 요약입니다.

3. Problem Statement

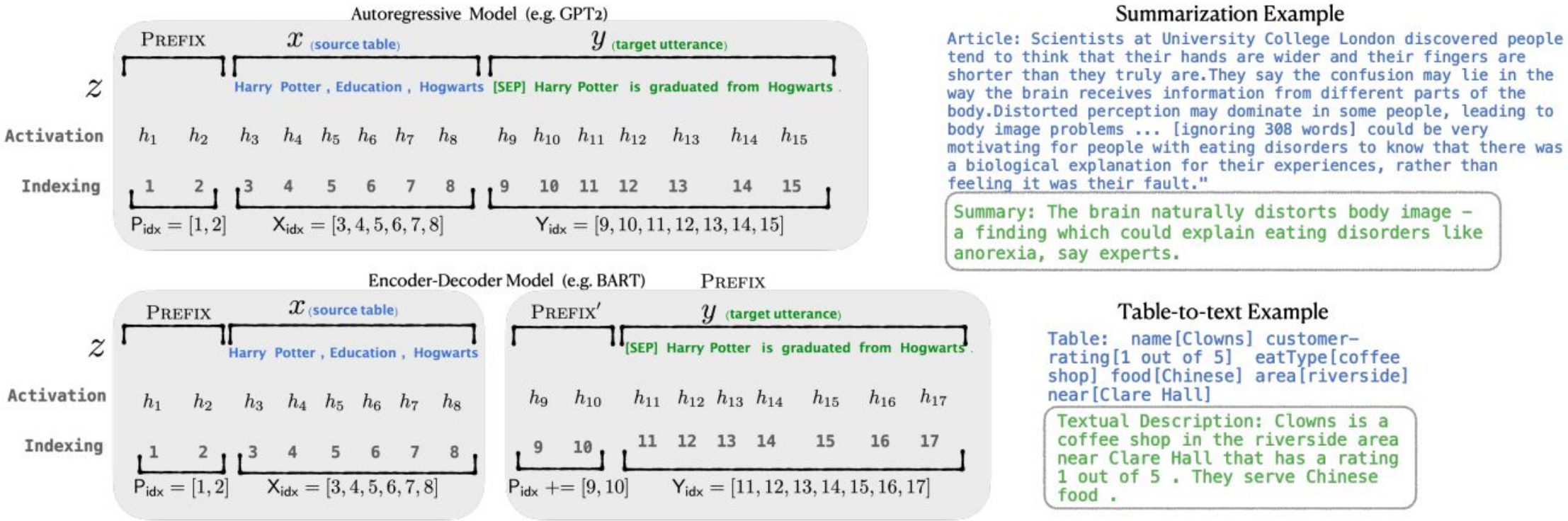

Figure 2: An annotated example of prefix-tuning using an autoregressive LM (top) and an encoder-decoder model (bottom). The prefix activations $\forall i \in \mathsf{P}_{\mathsf{idx}}, h_i$ are drawn from a trainable matrix P_θ. The remaining activations are computed by the Transformer.

3.1. Autoregressive LM

- Vaswani 등(2017)의 Transformer 아키텍처를 기반으로 한, φ로 매개변수화된 자동회귀 언어 모델 $p_\varphi(y \mid x)$를 가정합니다 (예: GPT-2; Radford 등, 2019).
- 그림 2(상단)에서 보여지는 것처럼, z = [x; y]는 x와 y의 연결이며; X_idx는 x에 해당하는 인덱스의 시퀀스를 나타내고, Y_idx는 y에 대해 동일하게 나타냅니다.
- 시간 스텝 i에서의 활성값(activation)는 $h_i \in R^d$ 입니다. 여기서 $h_i=[h_i^{(1)};\ldots;h_i^{(n)}]$은 이 시간 스텝에서의 모든 활성화 층의 연결이며, $h_i^{(j)}$는 시간 스텝 i에서 j번째 Transformer 층의 활성값(acitivation) 입니다.

3.1. Autoregressive LM

- 자동회귀 Transformer 모델은 h_i를 z_i와 그 왼쪽 컨텍스트의 과거 활성화 상태를 이용한 함수로 계산합니다. 이는 다음과 같습니다:

$$h_i = \mathrm{LM}_\phi(z_i, h_{<i}), \qquad (1)$$

- 여기서 h_i의 마지막 층은 다음 토큰의 분포를 계산하는 데 사용되며: $p_\phi(z_{i+1} | h_{\le i}) = \mathrm{softmax}(W_\phi h_i^{(n)})$이고, W_ϕ는 사전에 훈련된 행렬로, $h_i^{(n)}$을 어휘 집합 크기의 로짓으로 매핑합니다.

3.2 Encoder-Decoder Architecture

- 또한, 우리는 인코더-디코더 아키텍처(예: BART; Lewis 등, 2020)를 사용하여 $p_\phi(y|x)$를 모델링할 수 있습니다. 여기서 x는 양방향 인코더에 의해 인코딩되며, 디코더는 y를 자동회귀적으로(인코딩된 x와 그 왼쪽 컨텍스트에 기반하여) 예측합니다.
- 우리는 동일한 인덱싱 및 활성화 표기법을 사용합니다. 그림 2(하단)에 표시된 것처럼. 모든 $i \in X_idx$에 대한 h_i는 양방향 Transformer 인코더에 의해 계산되며; 모든 $i \in Y_idx$에 대한 h_i는 동일한 방정식(1)을 사용하여 자동회귀 디코더에 의해 계산됩니다.

3.3 Method: Fine-tuning

- 미세 조정(fine-tuning) 프레임워크에서, 우리는 사전 훈련된 매개변수 ϕ로 초기화합니다. 여기서 p_ϕ는 훈련 가능한 언어 모델 분포이고, 우리는 다음 로그 가능도(log-likelihood) 목표에 대해 경사(gradient) 업데이트를 수행합니다:

$$\max_{\phi} \log p_\phi(y \mid x) = \sum_{i \in Y_{idx}} \log p_\phi(z_i \mid h_{<i}). \quad (2)$$

4. Prefix-Tuning

- 우리는 조건부 생성 작업에 대해 세부 조정(fine-tuning)의 대안으로서 "prefix-tuning"을 제안합니다. 우리는 먼저 §4.1에서 직관을 제공한 뒤, §4.2에서 우리의 방법을 공식적으로 정의합니다.
- 참고로, § 기호는 "section"을 의미합니다. 따라서 §4.1, §4.2는 각각 섹션 4.1, 섹션 4.2를 지칭합니다.

4.1. Intuition

- 프롬프팅에서의 직관을 바탕으로, 우리는 적절한 컨텍스트를 가지면 언어 모델(LM)을 그 파라미터를 변경하지 않고도 이끌 수 있다고 믿습니다. 예를 들어, 만약 우리가 LM에게 단어(예를 들어, "Obama")를 생성하게 하고 싶다면, 우리는 그의 일반적인 연어(collocations)를 컨텍스트로 앞에 추가할 수 있습니다(예를 들어, "Barack"), 그리고 LM은 원하는 단어에 훨씬 높은 확률을 할당할 것입니다.
- 이 직관을 단일 단어나 문장을 생성하는 것을 넘어서 확장하면, 우리는 NLG 작업을 해결하기 위해 LM을 이끌 컨텍스트를 찾고자 합니다. 직관적으로, 컨텍스트는 x로부터 무엇을 추출할지를 지시함으로써 x의 인코딩에 영향을 미칠 수 있고; 그리고 다음 토큰 분포를 이끌어내어 y의 생성에 영향을 줄 수 있습니다.
- 그러나, 그러한 컨텍스트가 존재하는지는 명확하지 않습니다. 자연어 작업 지시사항(예를 들어, "다음 표를 한 문장으로 요약하시오")은 전문 주석자에게 작업을 해결하는 데 도움이 될 수 있지만, 대부분의 사전 훈련된 LM들에게는 실패할 수 있습니다.
- 데이터 기반의 이산 지침(discrete instructions)에 대한 최적화가 도움이 될 수 있지만, 이산 최적화는 계산상의 도전을 안고 있습니다.

4.1. Intuition

- 이산 토큰(discrete tokens)들에 대해 최적화를 수행하는 대신에, 우리는 지시사항을 연속적인 단어 임베딩(continous word embedding)으로 최적화할 수 있고, 그 효과는 모든 Transformer 활성화 레이어로 위쪽으로, 그리고 이어지는 토큰으로 오른쪽으로 전파될 것입니다.
- 이것은 실제 단어의 임베딩과 일치를 요구하는 이산 프롬프트보다 엄밀하게 표현력이 높습니다. 한편으로는, 이것은 긴 범위의 의존성을 피하고 더 많은 조절 가능한 파라미터를 포함하는 활성화 레이어의 모든 레이어를 조절하는 것보다 표현력이 낮습니다(§7.2). 따라서, prefix-tuning은 접두사의 모든 레이어를 최적화합니다.

4.2. Method

- Prefix-tuning은 z = [PREFIX; x; y]를 얻기 위해 자기회귀적인 LM(언어 모델)에 접두사(prefix)를 추가하거나, 그림 2에 보여진대로 z = [PREFIX; x; PREFIX'; y]를 얻기 위해 인코더와 디코더 양쪽에 접두사를 추가합니다.
- 여기서 P_idx 는 접두사 인덱스의 순열을 나타내고, 우리는 |P_idx|를 사용하여 접두사의 길이를 나타냅니다.

- 우리는 접두사가 자유 파라미터(free parameter)인 것을 제외하고는 식(1)에 있는 순환 관계를 따릅니다. Prefix-tuning은 차원이 |P_idx | × dim(h_i)인 훈련 가능한 행렬 P_θ(θ로 매개변수화됨)를 초기화하여 접두사 매개변수를 저장합니다.

$$h_i = \begin{cases} P_\theta[i,:], & \text{if } i \in \mathrm{P_{idx}}, \\ \mathrm{LM}_\phi(z_i, h_{<i}), & \text{otherwise.} \end{cases} \quad (3)$$

4.2. Method

- 훈련 목표는 식(2)와 동일하지만, 훈련 가능한 매개변수의 집합이 변경됩니다: 언어 모델 매개변수 φ는 고정되어 있고, 접두사 매개변수 θ만이 훈련 가능한 매개변수입니다.
- 여기서 h_i (모든 i에 대하여)는 훈련 가능한 P_θ의 함수입니다.
- i가 P_idx에 속하는 경우, 이것은 명확합니다. 왜냐하면 h_i는 P_θ에서 직접 복사하기 때문입니다.
- i가 P_idx에 속하지 않는 경우에도, h_i는 여전히 P_θ에 의존합니다. 왜냐하면 접두사 활성화는 항상 왼쪽 컨텍스트에 있으며 따라서 그 오른쪽에 있는 어떠한 활성화에도 영향을 미칠 것이기 때문입니다.

4.3. Parametrization of P_θ

- 실제로, P_θ 매개변수를 직접 업데이트하는 것은 불안정한 최적화와 약간의 성능 하락을 초래합니다.
- 따라서 우리는 행렬을 재매개변수화(reparameterization)합니다: P_θ[i,:]=MLP_θ(P'_θ[i,:]), 여기서 작은 매트릭스(P'_θ)가 큰 피드포워드 신경망(MLP_θ)과 합성됩니다.
- 주목할 점은 P_θ와 P'_θ이 같은 행의 차원(즉, 접두사의 길이)을 가지고 있지만, 다른 열의 차원(columns dimension)을 가지고 있다는 것입니다.
- 한번 훈련이 완료되면, 이러한 재매개변수화 매개변수는 생략될 수 있고, 접두사(P_θ)만 저장되면 됩니다.

6. Main Results

	E2E					WebNLG									DART					
	BLEU	NIST	MET	R-L	CIDEr	BLEU			MET			TER ↓			BLEU	MET	TER ↓	Mover	BERT	BLEURT
						S	U	A	S	U	A	S	U	A						
GPT-2$_{\text{MEDIUM}}$																				
FINE-TUNE	68.2	8.62	**46.2**	71.0	2.47	**64.2**	27.7	46.5	**0.45**	0.30	0.38	**0.33**	0.76	0.53	46.2	**0.39**	**0.46**	**0.50**	**0.94**	**0.39**
FT-TOP2	68.1	8.59	46.0	70.8	2.41	53.6	18.9	36.0	0.38	0.23	0.31	0.49	0.99	0.72	41.0	0.34	0.56	0.43	0.93	0.21
ADAPTER(3%)	68.9	8.71	46.1	71.3	2.47	60.4	**48.3**	54.9	0.43	**0.38**	**0.41**	0.35	**0.45**	**0.39**	45.2	0.38	**0.46**	**0.50**	**0.94**	**0.39**
ADAPTER(0.1%)	66.3	8.41	45.0	69.8	2.40	54.5	45.1	50.2	0.39	0.36	0.38	0.40	0.46	0.43	42.4	0.36	0.48	0.47	**0.94**	0.33
PREFIX(0.1%)	**69.7**	**8.81**	46.1	**71.4**	**2.49**	62.9	45.6	**55.1**	0.44	**0.38**	**0.41**	0.35	0.49	0.41	**46.4**	0.38	**0.46**	**0.50**	**0.94**	**0.39**
GPT-2$_{\text{LARGE}}$																				
FINE-TUNE	68.5	8.78	46.0	69.9	2.45	**65.3**	43.1	55.5	**0.46**	0.38	**0.42**	**0.33**	0.53	0.42	**47.0**	**0.39**	0.46	**0.51**	**0.94**	**0.40**
Prefix	**70.3**	**8.85**	**46.2**	**71.7**	**2.47**	63.4	**47.7**	**56.3**	0.45	**0.39**	**0.42**	0.34	**0.48**	**0.40**	46.7	**0.39**	**0.45**	**0.51**	**0.94**	**0.40**
SOTA	68.6	8.70	45.3	70.8	2.37	63.9	52.8	57.1	0.46	0.41	0.44	-	-	-	-	-	-	-	-	-

Table 1: Metrics (higher is better, except for TER) for table-to-text generation on E2E (left), WebNLG (middle) and DART (right). With only 0.1% parameters, Prefix-tuning outperforms other lightweight baselines and achieves a comparable performance with fine-tuning. The best score is boldfaced for both GPT-2$_{\text{MEDIUM}}$ and GPT-2$_{\text{LARGE}}$.

6. Main Results

Source	name : The Eagle | type : coffee shop | food : Chinese | price : cheap | customer rating : average | area : riverside | family friendly : no | near : Burger King
Prefix (50)	The Eagle is a cheap Chinese coffee shop located near Burger King.
Prefix (100)	The Eagle is a cheap coffee shop located in the riverside near Burger King. It has average customer ratings.
Prefix (200)	The Eagle is a cheap Chinese coffee shop located in the riverside area near Burger King. It has average customer ratings.
Prefix (500)	The Eagle is a coffee shop that serves Chinese food. It is located in the riverside area near Burger King. It has an average customer rating and is not family friendly.
FT (50)	The Eagle coffee shop is located in the riverside area near Burger King.
FT (100)	The Eagle is a cheap coffee shop near Burger King in the riverside area. It has a low customer rating and is not family friendly.
FT (200)	The Eagle is a cheap Chinese coffee shop with a low customer rating. It is located near Burger King in the riverside area.
FT (500)	The Eagle is a cheap Chinese coffee shop with average customer ratings. It is located in the riverside area near Burger King.

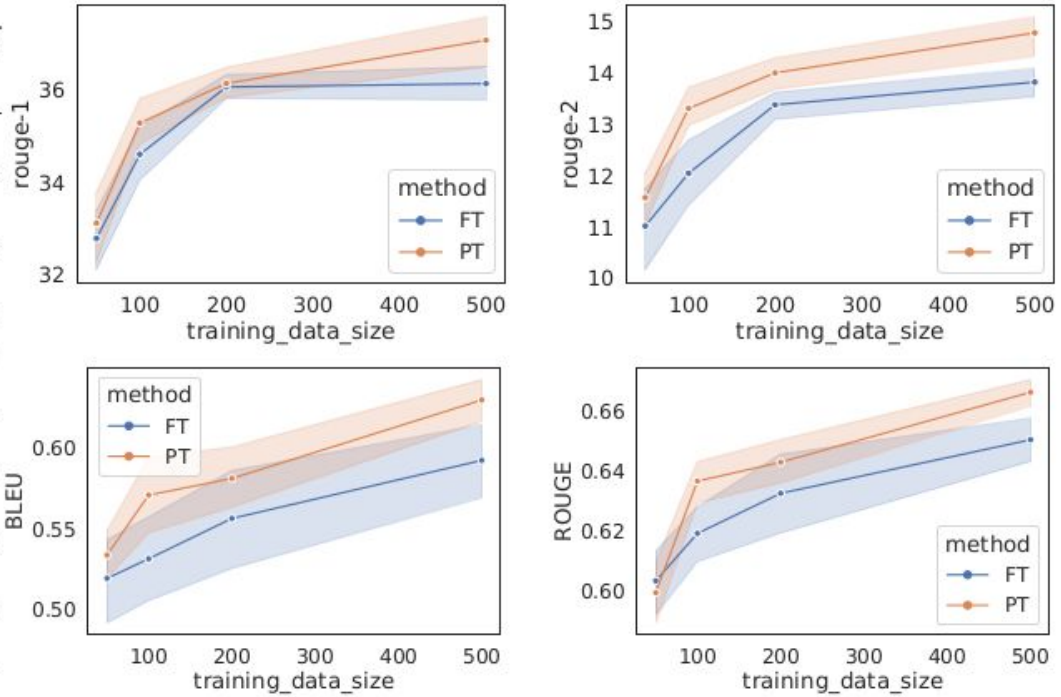

Figure 3: (Left) qualitative examples in lowdata settings. (Right) prefix-tuning (orange) outperforms fine-tuning (blue) in low-data regimes in addition to requiring many fewer parameters. The top two plots correspond to summarization, measured by ROUGE-1 and ROUGE-2. The bottom two plots correspond to table-to-text, measured by BLEU and ROUGE-L. The x-axis is the training size and the y-axis is the evaluation metric (higher is better).

6. Main Results

	R-1 ↑	R-2 ↑	R-L ↑
FINE-TUNE(Lewis et al., 2020)	45.14	22.27	37.25
PREFIX(2%)	43.80	20.93	36.05
PREFIX(0.1%)	42.92	20.03	35.05

Table 2: Metrics for summarization on XSUM. Prefix-tuning slightly underperforms fine-tuning.

	news-to-sports			within-news		
	R-1 ↑	R-2 ↑	R-L ↑	R-1 ↑	R-2 ↑	R-L ↑
FINE-TUNE	38.15	15.51	30.26	39.20	16.35	31.15
PREFIX	39.23	16.74	31.51	39.41	16.87	31.47

Table 3: Extrapolation performance on XSUM. Prefix-tuning outperforms fine-tuning on both news-to-sports and within-news splits.

6. Main Results

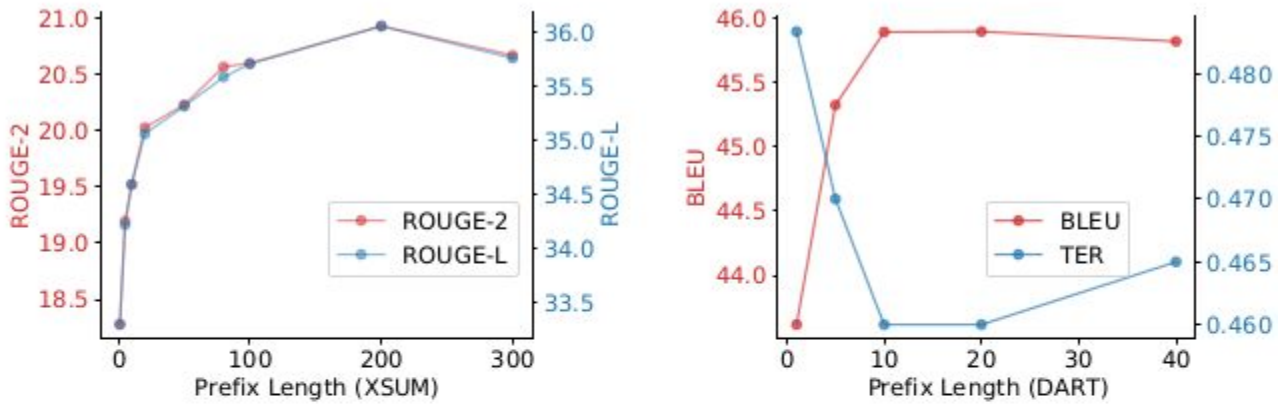

Figure 4: Prefix length vs. performance on summerization (left) and table-to-text (right). Performance increases as the prefix length increases up to a threshold (200 for summarization and 10 for table-to-text) and then a slight performance drop occurs. Each plot reports two metrics (on two vertical axes).

9. Conclusion

- 우리는 NLG 작업을 위한 훈련 가능한 연속적인 접두사를 추가하는 미세 조정의 경량화된 대안인 접두사 조정을 제안했습니다.
- 미세 조정보다 1000배 적은 매개변수만을 학습함에도 불구하고, 접두사 조정은 전체 데이터 설정에서 비교 가능한 성능을 유지할 수 있으며, 적은 데이터의 추론 설정에서 미세 조정을 능가합니다.