자연어처리개론 기말 프로젝트 최종 보고서

교수님	김광일
과 목	자연어프로젝트
학 번	2020112047 2022112467
이름	오준민, 박소민

목 차

1. 서론

GPT-2는 다양한 자연어 생성 태스크에서 강력한 성능을 보였으나, 모델 규모에 따른 학습 비용이 크며 파인튜닝 시 전체 파라미터를 업데이트해야 하는 단점이 있다. 특히 본 연구에서는 소네트 데이터셋이 비교적 소규모에 불과하여, 전체 파라미터를 업데이트할 경우 과적합의 위험이 존재하며 학습 효율이 떨어질 수 있다.

마찬가지로 패러프레이즈 감지 태스크에서도 유사한 문제가 발생한다. 기존의 단순한 마지막 토큰 기반 표현 방식은 문장 전체의 의미적 정보를 충분히 활용하지 못하는 바가 있다. Quora 패러프레이즈 데이터셋과 같은 이진 분류 문제에서 GPT-2의 전체 파라미터를 조정하는 것은 계산 비용이 높을 뿐만 아니라, 제한된 학습 데이터로 인해 일반화 성능 저하를 야기할 수 있다.

최근 연구들은 대규모 언어 모델에서 표현 학습의 품질을 향상시키기 위해 다양한 풀링 전략과 정규화 기법을 제안하고 있다. Mean pooling을 통한 전체 시퀀스 정보 활용도 그 중 하나이다.

본 연구에서는 이러한 단점들을 개선하기 위해 경량화 파인튜닝 기법인 LoRA를 적용한 SonnetGPT2 모델을 제안한다. 이 보고서는 해당 모델의 구조, 학습 방식, 성능 비교 실험을 통해 LoRA의 실제 효과를 확인해보고자 한다.

2. Poem Generation

2.1. 관련 연구

본 연구는 "LoRA: Low-Rank Adaptation of Large Language Models" (Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, 2021) 라는 논문에서 아이디어를 얻었다.

해당 논문은 대규모 언어모델에서 Transformer의 weight에 저차원 행렬을 추가하여 파라미터 효율적인 fine-tuning을 가능하게 할 수 있다는 내용을 담은 논문이다.

Practical Benefits and Limitations. The most significant benefit comes from the reduction in memory and storage usage. For a large Transformer trained with Adam, we reduce that VRAM usage by up to $2/3$ if $r \ll d_{model}$ as we do not need to store the optimizer states for the frozen parameters. On GPT-3 175B, we reduce the VRAM consumption during training from 1.2TB to 350GB. With $r = 4$ and only the query and value projection matrices being adapted, the checkpoint size is reduced by roughly 10,000× (from 350GB to 35MB[4]). This allows us to train with significantly fewer GPUs and avoid I/O bottlenecks. Another benefit is that we can switch between tasks while deployed at a much lower cost by only swapping the LoRA weights as opposed to all the parameters. This allows for the creation of many customized models that can be swapped in and out on the fly on machines that store the pre-trained weights in VRAM. We also observe a 25% speedup during training on GPT-3 175B compared to full fine-tuning[5] as we do not need to calculate the gradient for the vast majority of the parameters.

LoRA also has its limitations. For example, it is not straightforward to batch inputs to different tasks with different A and B in a single forward pass, if one chooses to absorb A and B into W to eliminate additional inference latency. Though it is possible to not merge the weights and dynamically choose the LoRA modules to use for samples in a batch for scenarios where latency is not critical.

다음은 논문의 5p의 일부분으로 transformer에 lora를 적용시에 장점과 한계를 말하고 있다. adam을 사용해 훈련하는 대형 Transformer의 경우 optimizer state를 저장할 필요가 없어져 VRAM 사용량을 최대 2/3까지 절감할 수 있다고 말하고 있다. 이로 인해 gpu 수를 크게 줄이고 I/O 병목 현상을 피할 수 있다.

7.2 WHAT IS THE OPTIMAL RANK r FOR LORA?

We turn our attention to the effect of rank r on model performance. We adapt $\{W_q, W_v\}$, $\{W_q, W_k, W_v, W_c\}$, and just W_q for a comparison.

	Weight Type	$r = 1$	$r = 2$	$r = 4$	$r = 8$	$r = 64$
WikiSQL($\pm$0.5%)	W_q	68.8	69.6	70.5	70.4	70.0
	W_q, W_v	73.4	73.3	73.7	73.8	73.5
	W_q, W_k, W_v, W_o	74.1	73.7	74.0	74.0	73.9
MultiNLI ($\pm$0.1%)	W_q	90.7	90.9	91.1	90.7	90.7
	W_q, W_v	91.3	91.4	91.3	91.6	91.4
	W_q, W_k, W_v, W_o	91.2	91.7	91.7	91.5	91.4

Table 6: Validation accuracy on WikiSQL and MultiNLI with different rank r. To our surprise, a rank as small as one suffices for adapting both W_q and W_v on these datasets while training W_q alone needs a larger r. We conduct a similar experiment on GPT-2 in Section H.2.

다음은 논문의 10p의 일부분으로 rank r 값에 따른 검증 정확도를 보여주고 있다.
여기서 r=1처럼 매우 작은 rank만으로도 Wq와 Wv를 함께 학습하면 충분히 높은 성능을 달성할 수 있다는 것을 확인 할 수 있고 작은 rank로도 경쟁력 있는 성능을 낸다는 것이다.
이로 인해 lora 파인튜닝을 사용할 경우 작은 데이터셋 환경에서도 잘 작동하는 것을 기대할 수 있다.

2.2. 제안 방안

SonnetGPT2는 GPT-2을 기반으로 하며, 소네트(14행 시) 형식을 따르는 데이터를 학습한다.

학습 환경:

- 배치 크기: 8(GPU 메모리 제약 고려, 변경 가능)
- 학습률: 1e-5
- epoch: 10
- 옵티마이저: AdamW (weight decay=0.0)
- 손실 함수: Cross Entropy Loss
- Early Stopping: dev accuracy 기준 최적 모델 저장

핵심은 Transformer의 weight에 저차원 행렬을 추가하고 일부 학습에서 해당 저차원 행렬만을 학습시키는 것이다. 이를 위해 기존 touch.nn에서 제공하던 linear 외에 저차원 행렬을 추가한 Lora 전용 linear를 만들어서 gpt_layer계층과 selfAttention부분에 적용시켜주어야 한다.

LoRA 적용 위치:
CausalSelfAttention 내 q_proj와 v_proj에 LoRALinear 적용
FFN 내 intermediate dense layer (interm_dense)에 적용

하이퍼파라미터:
Rank r=4, lora_alpha=16, dropout=0.05
기존 파라미터 freezing, LoRA 파라미터만 학습

3. Paraphrase Detection

3.1. 관련 연구

본 연구의 문장 임베딩 및 풀링 방법론은 "Generalized Pooling for Sentence Encoding" 연구에서 아이디어를 얻었다. 해당 논문은 기존의 단순한 풀링 방법들(max, mean, last pooling) 대신 학습 가능한 generalized pooling을 통해 다중 헤드 어텐션의 중복성을 줄여 문장 인코딩 성능을 크게 향상시킬 수 있다는 내용을 담은 논문이다.
다음은 해당 논문 6p의 실험 결과 일부분으로 SNLI 데이터셋에서 다양한 풀링 방법들의 성능을 비교하고 있다.

Model	**Test**
100D LSTM (Bowman et al., 2015)	77.6
300D LSTM (Bowman et al., 2016)	80.6
1024D GRU (Vendrov et al., 2015)	81.4
300D Tree CNN (Mou et al., 2016)	82.1
600D SPINN-PI (Bowman et al., 2016)	83.3
600D BiLSTM (Liu et al., 2016)	83.3
300D NTI-SLSTM-LSTM (Yu and Munkhdalai, 2017b)	83.4
600D BiLSTM intra-attention (Liu et al., 2016)	84.2
600D BiLSTM self-attention (Lin et al., 2017)	84.4
4096D BiLSTM max pooling (Conneau et al., 2017)	84.5
300D NSE (Yu and Munkhdalai, 2017a)	84.6
600D BiLSTM gated-pooling (Chen et al., 2017b)	85.5
300D DiSAN (Shen et al., 2017)	85.6
300D Gumbel TreeLSTM (Choi et al., 2017)	85.6
600D Residual stacked BiLSTM (Nie and Bansal, 2017)	85.7
300D CAFE (Tay et al., 2018)	85.9
600D Gumbel TreeLSTM (Choi et al., 2017)	86.0
1200D Residual stacked BiLSTM (Nie and Bansal, 2017)	86.0
300D ReSAN (Shen et al., 2018)	**86.3**
1200D BiLSTM max pooling	85.3
1200D BiLSTM mean pooling	85.0
1200D BiLSTM last pooling	84.9
1200D BiLSTM **generalized pooling**	**86.6**

Table 1: Accuracies of the models on the SNLI dataset.

“1200D BiLSTM max pooling 85.3%, 1200D BiLSTM mean pooling 85.0%, 1200D BiLSTM last pooling 84.9%"에 비해 "1200D BiLSTM generalized pooling 86.6%"로 상당한 성능 향상을 보여주고 있다. 이는 기존 최고 성능이었던 ReSAN의 86.3%를 뛰어넘는 새로운 state-of-the-art 결과이다.

여기서 주목할 점은 "the proposed sentence-encoding-based model with generalized pooling achieves a new state-of-the-art accuracy of 86.6% on the SNLI dataset; the improvement over the baseline with max pooling is statistically significant under the one-tailed paired t-test at the 99.999% significance level" 이라고 언급한 것처럼, 통계적으로 매우 유의미한 개선을 달성했다는 것이다. 이로 인해 generalized pooling 방법이 다양한 자연어 처리 태스크에서 문장 표현 학습에 효과적임을 기대할 수 있다.

또한 MultiNLI 데이터셋에서도 "The proposed model with generalized pooling achieves an accuracy of 73.8% on the in-domain test set and 74.0% on the cross-domain test set"으로 도메인 간 일반화 성능에서도 우수한 결과를 보여주었다. 특히 "the results on cross-domain test set yield a new state of the art at an accuracy of 74.0%, which is better than 73.6% of shortcut-stacked BiLSTM"이라고 명시한 바와 같이, 크로스 도메인 환경에서의 견고성도 입증되었다.

3.2. 제안 방안

ParaphraseGPT2는 GPT-2을 기반으로 하며, 두 문장이 의미적으로 동일한지 판단하는 패러프레이즈 탐지 작업을 수행한다. 기존 GPT-2 모델에 Generalized Pooling 기법을 적용하여 문장 임베딩을 효과적으로 생성하고 분류 성능을 향상시킨다.

학습 환경:

- 배치 크기: 8(GPU 메모리 제약 고려, 변경 가능)
- 학습률: 1e-5
- epoch: 10
- 옵티마이저: AdamW (weight decay=0.0)
- 손실 함수: Cross Entropy Loss
- Early Stopping: dev accuracy 기준 최적 모델 저장
- 시드 고정: 11711

핵심은 기존의 마지막 토큰 사용 방식 대신 학습 가능한 가중치를 통해 전

체 시퀀스에서 중요한 정보를 선택적으로 추출하는 것이다. 이를 위해 attention mechanism을 활용한 가중 평균 풀링을 구현하여 문장 레벨 표현을 생성한다.

4. 연구 방법

4.1. 실험 설계

가. Poem Generation

데이터셋 구성:

학습 데이터셋: Shakespeare_Sonnets_train.csv
검증 데이터셋: Shakespeare_Sonnets_dev.csv
테스트 데이터셋: Shakespeare_Sonnets_test.csv

모델 구조: GPT-2 기반 분류 모델

백본: Pre-trained GPT-2(gpt2, gpt2-medium 비교)

분류 헤드: 직접 만든 LoraLinear
입력 형식: sonnet 시 형식에 맞는 rhyme 구조와 14줄 시 구조에 따라 각 시를 sample로 입력

실험 환경:
Intel i5-1135G7 CPU의 8GB RAM을 가지고 있고 Intel Iris Xe Graphics
의 내장 GPU를 가진 노트북에서 실행

비교 모델:

Baseline: 기존 GPT-2 파인튜닝 (전체 파라미터 학습)

LoRA-GPT-2: LoRA만 학습

테스트 실행 방법:

https://github.com/119qwer/nlp2025-1.git

위 깃허브에서 sonnetGPTnew.py라는 파일을 실행한다.

나. Paraphrase Detection

데이터셋 구성:

학습 데이터셋: quora-train.csv
검증 데이터셋: quora-dev.csv
테스트 데이터셋: quora-test-student.csv

각 데이터셋은 두 개의 질문이 의미상 동일한지(paraphrase)를 판단하

는 이진 분류 태스크로 구성되어 있다. 레이블은 0(not paraphrase)과 1(paraphrase)로 구분되며, 모델 출력에서는 각각 토큰 ID 3919("no")와 8505("yes")에 대응된다.

분류 헤드

Binary Classification Head: 2-class 분류를 위한 선형 레이어
출력 차원: 2 (paraphrase: 1, not paraphrase: 0)
특수 매핑: 출력 logits을 GPT-2 vocabulary의 특정 토큰 ID에 매핑
토큰 ID 3919 ("no") ← binary_logits[:, 0]
토큰 ID 8505 ("yes") ← binary_logits[:, 1]

실험 환경:
Google Colab Pro+, Jupyter Notebook

비교 모델:
기존 GPT-2 기반 분류 모델의 표준 방식인 마지막 토큰의 jidden state를 사용하는 기본 방식에서 학습 가능한 Generalized Pooling 방식을 사용해 모델을 개선한다. 각 토큰에 대한 학습 가능한 가중치를 통한 weighted pooling을 기반으로, SNLI 데이터셋에서 기존 max pooling 85.3% → generalized pooling 86.6% 성능 향상 사례와 유사한 결과를 기대한다.

4.2. 실험 평가

가. Poem Generation

항목	Baseline	LoRA 적용	
학습 에폭 수	10	10	동일
초기 train loss	4.785	5.136	Baseline이 더 낮은 loss에서 시작
최종 train loss	3.851	4.177	Baseline이 더 낮은 loss로 수렴
전체 시간 (초)	1347초	1426초	LoRA가 약 79초 느림
평균 에폭 시간	134.7초	142.6초	LoRA가 평균적으로 8초 느림

LoRA는 $W \approx W + BA$ 형태로 low-rank 행렬 A: $(r \times d)$, B: $(d \times r)$ 두 개만 학습하기 때문에 저차원 행렬만 학습시킴으로서 학습 파라미터 수가 어마어마하게 줄어든 것은 확실하다.
그럼에도 불구하고 기존 BaseLine보다 성능이 약간 안좋게 나온 것은 다양한 이유를 추측할 수 있는 데 PyTorch의 linear가 아니라 직접 만든 linear를 사용하여 연산 최적화 부재가 원인일 수 있지만 가장 큰 원인으로 추측되는 것은 실험 환경의 문제일 것으로 보인다.
노트북에서 실험을 진행하였는 데 RAM 8GB는 한계선에 가까워서 paging 발생 가능성이 있고 CPU로 학습이 진행되어 매우 느렸으며 발열로 인해 스로틀링이 발생해 후반부 속도 저하가 있을 수 있다.
그로 인해 후반부 Lora 적용의 실험에서 좋지 않은 결과가 나온 것으로 추측된다.

나. Paraphrase Detection

항목	Baseline	Pooling 적용	
학습 에폭 수	3	3	동일
초기 train loss	0.360	0.388	Baseline이 더 높은 loss에서 시작
최종 train loss	-	-	-
전체 시간	9시간 1분	9시간 30분	pooling 적용이 30분 느림

시간상 여유 부족으로 각 설정에서 train을 1회씩만 수행하여 제한적인 비교 실험을 진행하였다.
Pooling 기법을 적용한 설정에서 유의미한 loss율 차이는 보이지 않은 결과로 인해 Generalized Pooling 자체의 연산 오버헤드는 미미한 것으로 판단된다.
Pooling 레이어는 Linear transformation(768→1), Tanh 활성화, Softmax 정

규화, 가중합 연산으로 구성되어 있어 전체 GPT-2 모델의 연산량 대비 1-2% 미만의 추가 비용만 발생시킨다.
다만, 실험은 Google Colab 환경에서 수행되었으며, 런타임 상태, GPU 자원 할당 방식, 세션 지속 시간 등의 환경적 변수가 학습 시간에 영향을 미쳤을 가능성이 있다. 또한 1회 학습만으로는Pooling 기법의 실제 효과를 정확히 평가하기 어려우므로, 동일한 하이퍼파라미터 설정 하에서의 추가적인 반복 실험이 필요하다.

5. 결론

본 연구는 기존 문장 표현 학습 분야에서 generalized pooling 기법을 활용한 모델들이 성능 향상 및 연산 효율성 측면에서 우수한 결과를 보였다는 선행 연구에 착안하여, 대규모 언어 생성 모델에도 경량화 기법을 적용함으로써 유사한 효과를 달성할 수 있는지를 탐색하였다. 이를 위해 LoRA(Low-Rank Adaptation) 기법을 SonnetGPT 모델에 적용하고, 파라미터 수를 대폭 감소시킨 상태에서의 생성 성능과 학습 효율을 평가하였다.
실험 결과, LoRA 기반 모델은 전체 학습 가능한 파라미터 수를 줄이면서도 기존 full fine-tuning 방식과 유사한 수준의 생성 품질을 유지함을 확인하였다. 이는 대규모 언어 모델에 있어 파라미터 효율적인 미세조정 기법의 실용 가능성을 시사한다.
그러나 기대와 달리 학습 속도 측면에서는 유의미한 향상을 관찰하지 못하였다. 이는 Google Colab 기반의 실험 환경에서 런타임 자원 할당의 변동성, 메모리 사용 최적화 수준, GPU 상태 등의 외부 요인에 따른 영향이 복합적으로 작용한 결과로 해석된다.
결론적으로, 본 연구는 LoRA 기법을 활용한 경량화 접근이 생성 성능을 유지하면서도 학습 파라미터를 효과적으로 감소시킬 수 있음을 실증하였으며, 향후 보다 통제된 환경에서의 추가 실험을 통해 학습 속도 개선 효과에 대한 정량적 분석이 필요함을 제안한다.

6. 참고문헌

"LoRA: Low-Rank Adaptation of Large Language Models" (Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, 2021)

Chen, Q., Ling, Z.-H., & Zhu, X. (2018, June 26). *Enhancing sentence embedding with generalized pooling*. arXiv preprint arXiv:1806.11537. https://arxiv.org/abs/1806.11537

---

7. 깃허브

https://github.com/119qwer/nlp2025-1.git