What Changes Can Large-scale Language Models Bring?

#	1
☑ Read	☑
☑ Review	☐
URL	https://arxiv.org/abs/2109.04650

What Changes Can Large-scale Language Models Bring? Intensive Study on HyperCLOVA : Billion-scale Korean Generative Pretrained Transformers

0. Abstract

HyperCLOVA

- 5600억의 토큰들로 이루어진 한국어 중심 말뭉치로 학습된 GPT-3의 한국어 버젼
- 한국어 태스크에서의 in-context zero-shot과 few-shot learning 성능에서 SOTA를 보임

No Code AI paradigm

- 프롬프트 엔지니어링 파이프라인에서의 융합
- 하이퍼클로바 스튜디오 : 인터랙티브 프롬프트 엔지니어링 인터페이스로, 머신러닝 비전문가들에게도 AI 프로토타이핑 능력 기능을 통한 노코드 패러다임 소개

1. Introduction

GPT-3 사용의 문제점

- 학습된 데이터의 **92.7%가 영어**로 이루어짐

    - 다른 언어적 특징 때문에, 다른 언어로 비슷한 모델들을 학습 할 때의 정보가 많이 없음
    - 기존에 제안 된 방법론이 어디에서는 적용되고 어디서는 실패할 지 모름
- 거대 언어 모델들의 작동 비용을 무시할 수 없는 입장에서 우리는 **130억, 1750억 크기의 모델**에 대한 정보 밖에 없음
- 입력값의 backward gradient가 필요한 **심화된 프롬프트 기반 학습 방법론**들은 in-context 거대 언어 모델에 실험을 해본 적이 없음
    - 예) 연속적인 프롬프트 기반 튜닝

이 논문의 제안

1. 우리는 거의 1000억 개의 파라미터에 다다르는 크기의 in-context 학습 기반 거대 언어 모델인 하이퍼클로바를 소개한다.
2. in-context 거대 언어 모델에 비영어권 언어의 말뭉치를 학습시켰을 때의 특정 언어별 토큰화 방법의 효과를 소개한다.
3. 39B 및 82B 개의 파라미터를 사용하는 중형 하이퍼클로바의 zero-shot 및 few-shot 기능을 살펴보고, 프롬프트 기반 튜닝에서 역방향 인풋이 가능할 때 다운스트림 태스크에서 SOTA임을 확인한다.
4. 하이퍼클로바 스튜디오를 사내 애플리케이션에 설계 및 적용하여 No Code AI의 실현 가능성을 제시한다. 입력 단계별 학습, 출력 필터, 그리고 지식 주입 기능을 갖춘 하이퍼클로바 스튜디오를 출시할 예정이다.

2. 선행 연구

2.1 프롬프트 최적화

- 언어 모델의 규모가 커짐에 따라 시공간 복잡도 측면에서 모델의 파인튜닝을 진행하는 것보다는 프롬프트 기반 접근 방식이 더 효율적임
- 동시에, 언어모델은 프롬프트 설계에 매우 민감하기에 프롬프트 최적화에 힘 써야함
- 프롬프트 최적화는 이산적 접근과 연속적 접근으로 나뉨
    - 이산적 접근 : 토큰 공간에서 직접 최적화를 하여 이전이 가능하나 해석에 어려움이 있으며 차선책일 뿐임
    - 연속적 접근

- 특정 다운스트림 태스크에 한해서는 자동 회귀 언어 모델에 대한 프롬프트 튜닝이 MLM 기반 파인튜닝보다 성능이 좋음

2.2 언어 모델

- 다국어 언어 모델이 공개적으로 제공되고 있지만, 여전히 각 언어 별 언어 모델에 대한 수요가 많으며, 비영어권 언어 모델은 비용이 높아 제한적임
- 비영어권 in-context 학습에 대한 연구도 활발하지 않으며, 다국어 in-context 학습에 대한 연구는 진행조차 되지 않았음
- 중국어 모델 관련해서는 관련 연구가 진행 중임

3. 사전 학습

3.1 데이터 설명

- Open AI GPT-3의 한국어 비율은 문자 수준으로 0.02%보다 낮음
- 아래 표에 나와 있는 데이터를 사용, 총 561B개의 토큰으로 이루어짐

Name	Description	Tokens
Blog	Blog corpus	273.6B
Cafe	Online community corpus	83.3B
News	News corpus	73.8B
Comments	Crawled comments	41.1B
KiN	Korean QnA website	27.3B
Modu	Collection of five datasets	6.0B
WikiEn, WikiJp	Foreign wikipedia	5.2B
Others	Other corpus	51.5B
Total		561.8B

Table 1: Descriptions of corpus for HyperCLOVA.

3.2 모델과 학습

- GPT-3과 같은 트랜스포머 디코더 구조 적용
- 연구는 되지 않았지만 많은 애플리케이션에서는 유용한 중간 규모의 파라미터로 이루어진 모델의 성능을 보여주고 싶어 아래 표와 같은 구성으로 실험 진행

# Param	n_{layers}	d_{model}	n_{heads}	d_{head}	lr
137M	12	768	16	48	6.0e-4
350M	24	1024	16	64	3.0e-4
760M	24	1536	16	96	2.5e-4
1.3B	24	2048	16	128	2.0e-4
6.9B	32	4096	32	128	1.2e-4
13B	40	5120	40	128	1.0e-4
39B	48	8192	64	128	0.8e-4
82B	64	10240	80	128	0.6e-4

Table 2: Detailed configuration per size of HyperCLOVA.

- 실험 설정
  - megatron-LM기반 모델
  - NVIDIA superpod : 1024개의 A100 GPU가 있는 128개의 DGX 서버
  - AdamW, 코사인 learning rate scheduler
  - batch-size : 1024
  - minimum LR : 1/10 of the original LR
- Scaling Law : 모델 사이즈를 늘리고 더 오래 학습 시키는 것이 성능이 좋음

3.3 한국어 토큰화

- 한국어는 교착어이기 때문에, 토큰화 방법이 한국어 언어 모델에 영향을 줌
- 형태소 기반 바이트 수준 바이트 페어 인코딩 사용(허깅페이스 토크나이저, MecabKo)

4. 실험 결과

4.1 실험 환경

- 크게 5개의 데이터셋으로 평가 진행함
- NSMC
  - 네이버 영화의 영화 리뷰 데이터셋
  - 70개의 예시가 있는 12개의 세트 구성

- KorQuAD 1.0
  - 한국어 버젼의 기계독해 데이터셋
  - 테스트 문단, 4개의 질문-답변 쌍, 그리고 테스트 질문으로 1번 실험
- AI Hub Korean-English
  - 한국어-영어 문장 쌍이 다양한 주제로 있는 데이터셋
  - 1000개의 쌍으로 한-영, 영-한 번역 실험 : BLEU 로 평가
- YNAT
  - 연합 뉴스 주제 분류 데이터셋
  - 3번의 in-context 70-shot 학습의 정확도를 평균 냄
- KLUE-STS
  - 문장 유사도 측정 데이터셋 : 0-5점 사이, F1도 사용함
  - 3번의 in-context 40-shot 학습으로 정확도 평균 냄
- Query modification task
  - AI 스피커를 위한 태스크로, 멀티 쿼리를 하나의 쿼리로 바꾸는 태스크
- Baseline 모델의 점수로 비교하였음

4.2 In-context Few-shot Learning

- 많은 in-context 학습 태스크들이 모델 사이즈가 증가함에 따라 성능도 증가하는 것을 알 수 있음

	NSMC (Acc)	KorQuAD (EM / F1)		AI Hub (BLEU) Ko→En	AI Hub (BLEU) En→Ko	YNAT (F1)	KLUE-STS (F1)
Baseline	89.66	74.04	86.66	40.34	40.41	82.64	75.93
137M	73.11	8.87	23.92	0.80	2.78	29.01	59.54
350M	77.55	27.66	46.86	1.44	8.89	33.18	59.45
760M	77.64	45.80	63.99	2.63	16.89	47.45	52.16
1.3B	83.90	55.28	72.98	3.83	20.03	58.67	60.89
6.9B	83.78	61.21	78.78	7.09	27.93	67.48	59.27
13B	87.86	66.04	82.12	7.91	27.82	67.85	60.00
39B	87.95	67.29	83.80	9.19	31.04	71.41	61.59
82B	88.16	69.27	84.85	10.37	31.83	72.66	65.14

Table 3: Results of in-context few-shot tasks on sentiment analysis, question answering, machine translation, topic classification, and semantic similarity per model size. As baselines, we report the results of BERT-base for NSMC and KorQuAD, and Transformer for AI Hub from Park et al. (2020). mBERT is used for YNAT and KLUE-STS from Park et al. (2021).

- 그러나 한-영 번역과 KLUE-STS 성능은 베이스라인에 비해 성능이 현저히 떨어짐
  - 한-영 번역의 성능 저하의 이유는 말뭉치 내에 영어 비율이 낮아서라고 추정함

4.3 프롬프트 기반 튜닝

- 4K의 예제로 프롬프트 튜닝을 하는 것이 RoBERTa를 150K의 데이터로 파인튜닝하는 것과 비슷한 결과를 냄
- 프롬프트 튜닝이 하이퍼클로바의 정확도 뿐만 아니라 **견고성을 증가**시킨다는 것을 알 수 있음

Methods	Acc
Fine-tuning	
mBERT (Devlin et al., 2019)	87.1
w/ 70 data only	57.2
w/ 2K data only	69.9
w/ 4K data only	78.0
BERT (Park et al., 2020)	89.7
RoBERTa (Kang et al., 2020)	91.1
Few-shot	
13B 70-shot	87.9
39B 70-shot	88.0
82B 70-shot	88.2
p-tuning	
137M w/ p-tuning	87.2
w/ 70 data only	60.9
w/ 2K data only	77.9
w/ 4K data only	81.2
13B w/ p-tuning	91.7
w/ 2K data only	89.5
w/ 4K data only	90.7
w/ MLP-encoder	90.3
39B w/ p-tuning	**93.0**

Table 4: Comparison results of p-tuning with fine-tuned LMs and in-context few-shot learning on NSMC. MLP-encoder means the result of replacing LSTM with MLP as the p-tuning encoder on 150K NSMC training data.

Model sizes	Few-shots	p-tuning	BLEU
13B	zero-shot	×	36.15
		O	**58.04**
	3-shot	×	45.64
		O	**68.65**
39B	zero-shot	×	47.72
		O	**73.80**
	3-shot	×	65.76
		O	**71.19**

Table 5: Results of p-tuning on in-house query modification task.

- 프롬프트 튜닝은 zero-shot과 3-shot 시나리오 모두에서 입력 쿼리 품질을 상당한 마진으로 개선함
- input 부분에서의 p-tuning을 적용한 첫 연구
- 이 방법론이 적용 가능하다면, large-scale GPU 클러스터 없이도 성능 향상 가능

4.4 토큰화의 효과

- 형태소 기반 바이트 수준 BPE
  - 대부분의 태스크에서 형태소 기반 BPE가 좋은 성능
  - 언어 특성별 토큰화 기법이 영향이 있다는 것을 알 수 있음
- 바이트 수준 BPE
- 문자 수준 BPE
  - OOV(out-of-vocabulary) 생성

	KorQuAD (EA / F1)		AI Hub (BLEU) Ko→En	AI Hub (BLEU) En→Ko	YNAT (F1)	KLUE-STS (F1)
Ours	**55.28**	**72.98**	3.83	**20.03**	**58.67**	**60.89**
byte-level BPE	51.26	70.34	**4.61**	19.95	48.32	60.45
char-level BPE	45.41	66.10	3.62	16.73	23.94	59.83

Table 6: Effects of tokenization approaches on five tasks. HyperCLOVA-1.3B is used for evaluation.

5. 산업에 끼치는 영향에 관한 논의

- NLP ML 운영의 라이프 사이클 가속화

5.1 하이퍼클로바 스튜디오

- OpenAI Playground와 같은 GUI 인터페이스를 제공
- 다양한 기능을 API 호출로 쉽게 출력을 얻을 수 있는 API 엔드포인트 지원

5.2 하이퍼클로바 스튜디오 사례 연구

- 목적 함수를 정의하거나 모델을 자동으로 평가하는 것이 간단함
- 입출력 스타일을 쉽게 제어 가능
- AI 지식이 없는 프로덕트 디자이너도 빠르고 쉽게 PoC 시스템 제작 가능

5.2.1 개성을 갖춘 챗봇의 신속한 프로토타이핑

- 캐릭터 속성에 대한 한두줄의 설명 및 몇 가지 대화 예제로 특정 페르소나를 가진 챗봇 구축 가능

5.2.2 Zero-shot transfer 데이터 증강

- 사용자의 의도가 들어간 문장을 입력하면, 그에 따른 발화를 생성해줌

5.2.3 이벤트 제목 생성

- 이커머스 플랫폼에서 상품 광고 강화를 위한 이벤트 제목 생성 작업
- 제품 특성을 설명하는 키워드를 인상적인 이벤트 제목으로 변환

5.3 하이퍼클로바 스튜디오의 기회

- 입력 그라데이션 API : 프롬프트 최적화를 통한 다운스트림 태스크 성능 향상
- 프롬프트 인젝션 모듈 : retriever가 검색한 적절한 문서를 사용
- 입출력 필터 : 오용 방지

5.4 No/Low Code AI 패러다임

- 일반적인 머신 러닝 개발 파이프라인
  - 문제 정의 및 사용자 조사 → 데이터 수집 및 주석 달기 → 모델 학습 및 검증 → MLOps 배포 및 운영 → 오류 분석 및 사용자 모니터링
- 위와 같은 단계를 완화시킴
  - No Code : PoC에서 빠른 반복이 유리하거나 대규모 모델의 순수 생성 기능만으로 서비스를 구축할 수 있는 경우에 유용
  - Low Code : 전처리 코드 또는 입출력 모듈이 필요한 경우 일부 학습 데이터셋을 사용하는 경우에 유용

6. 결론

- 한국에서 하이퍼클로바 스튜디오를 통해 머신러닝에 익숙하지 않은 사람들도 자신만의 AI를 구축할 수 있는 시스템을 만드는 것이 목표
- 대규모 언어 모델의 오용 : 루다의 악의적인 사용, 프라이버시 문제
- 공정성, 편견, 대표성 : 성별과 인종차별을 포함한 편향적 반응

- 과도한 에너지 소비
- 긍정적인 방향을 위한 노력 : 노/로우 코드의 가능성