💚

1. What Changes Can Large-scale Language Models Bring

0. Abstract

- 이 논문에서는 GPT-3 논문에서 **덜 다루어진 몇 가지 문제**를 다룸
  - 비영어(non-English) 언어 모델
  - 다양한 크기의 모델의 성능
  - 프롬프트 최적화가 문맥 내 학습에 미치는 영향

  ⇒ 이를 위해 **HyperCLOVA**를 소개
- **HyperClOVA**
  - 560B 토큰 규모의 한국 중심 말뭉치로 훈련된 82B GPT-3의 한국어 변형 모델
  - 한국어 특화 토큰화를 통해 문맥 내 zero-shot 및 few-shot 학습 성능에서 최신 기술을 보여줍니다
    - **내용 보충**

    ▼ zero-shot

    - 모델이 이전에 학습한 클래스나 작업에 대해 직접적인 학습 데이터를 가지고 있지 않을 때 새로운 클래스나 작업을 수행하는 능력을 의미
    - 모델이 추론 및 일반화 능력을 활용하여 새로운 정보를 이해하고 처리하는 데 중점을 둠
    - 주로 사전 학습된 언어 모델과 같은 큰 모델에서 발견되며, 문장 분류/번역/감정 분석과 같은 다양한 NLP 작업에서 적용
      - ex) "이것은 사과입니다."와 같은 설명을 사용하여 모델에게 "사과"와 관련된 작업을 수행하도록 지시할 수 있음

    ▼ few-shot

    - 모델이 매우 적은 양의 학습 데이터만을 사용하여 새로운 작업을 수행하는 능력을 의미

        - 주로 모델이 몇 가지 예시나 지침을 통해 작업을 이해하고 실행하는 데 사용
            - 일반적으로 작업 지시어나 목표 지시어와 함께 적은 수의 학습 샘플을 제공
            - ex) "Netflix에서 '호두까기 인형'과 관련된 영화를 추천하십시오."라는 지시를 통해 모델은 "호두까기 인형"과 관련된 영화를 추천하는 작업을 수행할 수 있음
        - 개별 사용자에 맞춤형 서비스 및 추천 시스템에서 특히 유용
    - 프롬프트 기반 학습 → 성능 측면에서의 이점
    - No Code AI 패러다임 구현 가능성을 논의 ⇒ **HyperCLOVA studio**

      → 비전문가에게 AI 프로토타이핑 능력을 제공

1. Introduction

- **선행 연구**
    - GPT-3: zero-shot 및 few-shot 성능 측면에서 높게 평가(Brown 등, 2020)
        - 문맥 내 학습 접근 방식에서는 자연어 작업 설명과 few-shot 예제로 구성된 이산 프롬프트가 대규모 언어 모델(LMs)을 제어하여 대상 작업에 대한 예측을 추론
    - OpenAI의 GPT-3를 사용하여 GPT-3의 문맥 내 학습 성능을 더 향상시킬 수 있는 방법을 제안(Zhao 등, 2021; Liu 등, 2021a)
    - 프롬프트 기반 학습 방법이 BERT, GPT-3 및 T5의 성능을 개선하는 데 파라미터 업데이트 없이 사용되었다고 보고(Liu 등, 2021b; Lester 등, 2021; Shin 등, 2020)
- 그러나 GPT-3 사용에 관련된 **세 가지 실용적 문제**가 있음
    1. 훈련 말뭉치의 언어 구성의 92.7%가 영어로 크게 편향되어 있음
        - 다른 언어의 작업에 적용하기 어려움
        - 다른 언어의 특성이 다른 비슷한 모델을 어떻게 훈련해야 하는지, 그리고 원래 제안된 방법이 어디에 자연스럽게 적용되고 어디에서 실패할 수 있는지에 대한 정보가 거의 x
    2. 중간 크기 모델에 대한 정보가 없음

    - 대규모 LMs 사용 비용을 고려하여 다양한 크기의 모델의 능력을 알아보는 것은 실용적이고 유용하지만, 13B(→ 소규모) 및 175B(→ 대규모) 모델에 대한 분석만 존재
3. 고급 프롬프트 기반 학습 방법은 아직 문맥 내 대규모 LM 학습에 실험되지 않음
    - 입력의 역방향 그래디언트를 필요로 하는 고급 프롬프트
    - 예시) 연속적인 프롬프트 기반 튜닝

- **HyperCLOVA**
    - 거의 100B 파라미터를 가진 대규모 한국어 문맥 내 학습 기반 LM
    - 560B 토큰의 대규모 한국어 중심 말뭉치를 구축하여 설계

      → 비영어 언어의 훈련 말뭉치에 대해 대규모 문맥 내 LM의 언어별 토큰화의 효과를 발견
    - 39B 및 82B 파라미터를 가진 중간 크기 HyperCLOVA의 zero-shot 및 few-shot 능력을 탐구
    - 입력의 역방향 그래디언트가 있는 경우 역방향 그래디언트를 사용할 때 다른 최신 모델을 능가하는 프롬프트 기반 튜닝이 성능을 향상시킬 수 있음을 발견
    - HyperCLOVA Studio를 설계하고 내부 응용 프로그램의 사용 가능성을 논의

      → 입력 그래디언트, 출력 필터 및 지식 주입과 같은 HyperCLOVA Studio의 기능을 공개할 예정

2. 선행 연구(Previous Work)

2-1. Prompt Optimization

- 프롬프트 기반 접근 방식
    - 언어 모델에게 최상의 지식을 유도하고 예측 성능을 극대화하기 위한 최적의 프롬프트를 구성하는 것
    - 언어 모델의 규모가 커짐에 따라 프롬프트 기반 접근 방식으로 전체 fine-tuning 패러다임을 대체할 수 있는 잠재력이 보고됨
    - 시간 및 공간 복잡성 측면에서 효율적
    - 하지만 언어 모델은 프롬프트 디자인에 매우 민감함

  ⇒ **프롬프트 최적화** 방법론 도모
- 프롬프트 최적화는 이산 및 연속 접근 방식으로 분류할 수 있음

    - 이산 접근 방식
        - 토큰 공간에서 직접 최적화
        - 전이성의 이점이 존재
        - 그러나 Shin 등(2020)은 이산 공간이 해석 가능성이 낮고 최적이 아닐 수 있다는 것을 보였
    - 연속 접근 방식
        - 메인 LM 파라미터를 파인튜닝하지 않고도 문맥화된 토큰 공간을 최적화하는 것을 제안(2021)
        - 자기 회귀형 LM에 대한 p-tuning이 일부 하위 작업에서 MLM 기반 fine-tuning보다 우수한 성능을 달성한다는 것을 발견(Liu 등, 2021b)
        - 잘 최적화된 프롬프트 기반 학습이 주요 벤치마크에서 최신 성능을 달성함(Lester 등, 2021)

2-2. 언어 모델들

- 다국어 언어 모델은 이미 공개됨
    - 하지만, **높은 비용** 때문에 영어 이외의 언어에 대한 언어별 언어 모델의 이용 가능성은 제한적임
- 문맥 내 학습자에 대한 연구는 주요 언어에만 집중되어 있음
- 최근에는 중국 말뭉치를 기반으로 한 GPT와 유사한 언어 모델이 활발하게 연구되고 있음
    - 중국어 말뭉치를 사용하여 2.6B 및 13B 파라미터를 가진 LM을 성공적으로 훈련시킴
    - 207B 모델, 해당 인프라 및 훈련 기술에 대한 연구를 진행중임

3. 사전 학습(pre-training)

3-1. Data Description

- OpenAI GPT-3에서의 한국어 데이터 비율은 문자 수로 측정하면 0.02% 미만으로 매우 작음
    - HyperCLOVA를 훈련하기 위해 사전에 대규모 한국어 중심 말뭉치를 구축하는 것이 매우 중요

- 대규모 말뭉치를 구축하기 위해 NAVER의 다양한 서비스와 외부 소스에서 사용자 생성 콘텐츠(UGC) 및 외부 파트너가 제공하는 내용을 포함한 모든 텍스트 데이터를 수집

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

- 데이터셋을 정제하고 총 561B 토큰을 최종 말뭉치로 수집
  - 말뭉치는 사전 훈련을 위해 무작위로 샘플링 됨

3-2. 모델 & 학습

- OpenAI의 GPT-3와 동일한 **Transformer Decoder** 아키텍처를 사용
  - 13B에서 175B의 OpenAI GPT-3로 가는 거의 지수적 보간을 설정

# 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

- megatron-LM을 기반으로 함
- NVIDIA Superpod에서 훈련됨

    - 1,024개의 A100 GPU를 가진 128개의 강력하게 클러스터링된 DGX 서버가 포함됨
- optimizer: AdamW
- 코사인 학습률 스케줄링과 가중치 감소 사용
- 미니배치 크기: 1024, 최소 학습률: 원래 학습률의 1/10
- 섹션 4의 실험에서는 모든 모델이 동일한 반복에서 훈련을 마치지 않았기 때문에 150B로 훈련된 모델을 사용
    - 섹션 5.2의 실험에서는 300B 토큰으로 훈련된 모델을 사용
- HyperCLOVA 말뭉치에 포함되지 않은 백과사전 말뭉치에서의 테스트 손실에서도 이전 연구에서 발견된 것처럼 스케일링 법칙을 관찰할 수 있음
    - 모델 크기를 늘리고 더 오래 훈련하면 이점이 있는 것을 확인할 수 있음

3-3. 한국어 Tokenizer

- 한국어는 명사 다음에 조사가 오고, 동사 또는 형용사의 어간 다음에 어미가 오는 접속 언어
    - 명사와 조사, 어간과 어미를 정확하게 토큰화하면 각 토큰의 의미를 명확히 할 수 있음
    - 한국어 LM에 적합한 정교한 토큰화 전략을 디자인해야 하며, 이는 영어와는 다름
- **HyperCLOVA**는 형태소를 고려한 바이트 수준 BPE를 토큰화 방법으로 사용

▼ 바이트 수준 BPE

- 바이트 페어 인코딩 (Byte Pair Encoding, BPE)
    - 자연어 처리와 데이터 압축 분야에서 널리 사용되는 텍스트 압축 및 인코딩 알고리즘 중 하나
    - 기본적으로 텍스트의 단어나 서브워드를 효율적으로 표현하기 위한 방법 중 하나로, 텍스트를 작은 단위로 나누고 가장 빈번하게 나타나는 바이트 쌍을 찾아서 새로운 서브워드로 대체하는 과정을 반복하여 작동
    - 기계 번역, 자연어 이해, 언어 생성 및 텍스트 분류 등 다양한 자연어 처리 작업에서 유용하게 활용
- BPE의 **주요 단계**
    1. 시작: 초기에는 모든 문자나 바이트가 서브워드로 간주됨

2. 학습 데이터: 학습 데이터에 있는 텍스트를 바이트 레벨로 나누고, 빈도가 가장 높은 바이트 쌍을 찾음
3. 새로운 서브워드 생성: 빈도가 가장 높은 바이트 쌍을 하나의 새로운 서브워드로 결합하고, 이를 텍스트에서 찾아서 대체
4. 반복: 위의 단계를 원하는 횟수만큼 반복, 일반적으로는 특정 횟수 또는 특정 크기의 어휘 사전을 생성할 때까지 반복됨

- BPE의 장점
  - 서브워드 분리: BPE는 단어를 서브워드로 분리하여 언어의 복잡성을 줄일 수 있습니다. 이를 통해 언어의 다양한 형태를 다루기 쉽게하고 언어 이해 작업의 성능을 향상시킬 수 있습니다.
  - 미지의 단어 처리: BPE를 사용하면 이전에 본 적이 없는 단어나 언어에 대한 작업에서도 유용하게 사용할 수 있습니다.
  - 효율적인 인코딩: BPE는 텍스트를 더 효율적으로 압축하고 표현하는 데 도움을 줍니다.
- GPT-2와 GPT-3도 바이트 수준 BPE를 사용
- 한국어는 영어와는 달리 'ㅎ', '하', 또는 '한'와 같은 비영어 문자는 모두 세 가지 다른 유니코드 바이트로 분리됨

⇒ **형태소 분석기**를 적용하여 바이트 수준 BPE의 문제를 완화

- **HyperCLOVA**는 공백과 내부 형태소 분석기에서 얻은 형태소를 사용하여 문장을 사전 분할함
  - 대부분의 비한국 문자를 제외
  - 형태소 분석기로 미리 분할된 문장을 사용하여 대부분의 비한국 문자가 단일 바이트 문자로 표현되는 문장을 학습
  - `HuggingFace`의 `tokenizers` 라이브러리를 사용

4. 실험 결과

4-1. 실험 설계

- 문맥 내 few-shot learning 성능을 평가하기 위해 5개(+1개)의 데이터셋을 사용
  - 하나의 **내부** 데이터셋
    - 프롬프트 기반 최적화 성능을 평가하기 위해

- **NSMC**
  - NAVER Movies에서 가져온 영화 리뷰 데이터셋
  - 15만 개의 훈련 데이터와 5만 개의 테스트 데이터를 포함하고 있음
  - few-shot 학습 실험을 위해 12개의 세트를 생성하며, 각 세트는 훈련 세트에서 무작위로 추출된 70개의 예제로 구성

    → 12개의 문맥 내 70-shot 학습 모델의 테스트 정확도를 평균
  - SST-2와 유사한 스키마
- **KorQuAD 1.0**
  - 기계 독해 데이터셋의 한국어 버전
  - 10645개의 훈련 단락, 66181개의 훈련 질문 및 5774개의 검증 질문으로 구성되어 있음
  - 데이터셋 형식은 SQuAD 1.0과 유사
  - SQuAD v2.0 평가 체계를 따름
    - 테스트 단락, 해당 샘플 네 개의 질문-답변 쌍 및 테스트 질문이 GPT-3에 입력됨

    ⇒ 단락 관점: zero-shot learner

    ⇒ 질문 관점: few-shot learner
  - 각 모델 크기에 대해 단일 시행을 수행
- **AI Hub 한-영 병렬 말뭉치**
  - 뉴스, 정부 웹사이트, 법적 문서 등에서 가져온 한-영 병렬 문장을 포함하고 있음
  - 80만 개의 문장 쌍이 포함되어 있으며, 한-영 및 영-한 번역 작업에서 1,000개의 문장 쌍을 무작위로 추출하여 평가
  - 각 번역 작업에 대해 세 번의 무작위 시행을 수행
  - 4-shot 학습에서 평가되며 각 시행에 대해 네 가지 다른 예제를 사용
  - 평가: BLEU 점수를 사용, Moses와 MeCab을 사용하여 비교

    ▼ BLEU 점수
    - BLEU (Bilingual Evaluation Understudy)

        - 자동 기계 번역 시스템의 출력을 평가하기 위한 널리 사용되는 평가 지표 중 하나
        - B두 가지 언어 간 번역 품질을 측정하는 데 사용되며, 주로 기계 번역 시스템의 성능을 평가하는 데 활용
        - BLEU 점수는 0에서 1 사이의 값을 가지며, 더 높은 점수는 더 좋은 번역 품질을 나타냄
    - **YNAT (Yonhap News Agency Topic Classification 또는 KLUE-TC)**
        - KLUE 벤치마크 작업 중 하나
        - 일곱 가지 클래스로 된 주제 분류 문제
        - 훈련, 검증 및 테스트 세트에 각각 4.5만, 9천 개 및 9천 개의 주제 헤드라인을 포함하고 있음
        - 3개의 문맥 내 70-shot 학습자의 테스트 정확도를 평균
    - **KLUE-STS**
        - KLUE 벤치마크의 또 다른 작업
        - 각 문장 쌍 간의 유사도를 예측하는 작업(점수: 0에서 5 사이의 값)
        - 실수 값 유사도를 이진화 한 후 F1 점수를 사용
        - 3개의 문맥 내 40-shot 학습자의 테스트 정확도를 평균
- 쿼리 수정 작업
    - AI 스피커 사용자를 위한 쿼리 수정 작업
    - 이미 AI 스피커에서 단일 턴 FAQ 시스템이 운영 중인 경우를 대상으로 함
    - 멀티턴 정보를 이해해야 하는 쿼리에서의 목표는 멀티턴 쿼리를 이해할 수 있는 단일턴 쿼리로 변환하는 것
    - 총 1,326개의 테스트 인스턴스가 있음
- Baseline
    - 문맥 내 학습 실험 ⇒ `BERT` 및 `Transformer`, `mBERT`의 점수를 사용
    - p-튜닝 실험 ⇒ `BERT-Multilingual` 및 `RoBERTa`를 사용

        ▼ p-튜닝

        자연어 처리 (NLP)에서 사전 학습된 언어 모델을 세부 작업(task-specific)에 맞게 미세 조정(fine-tuning)하는 과정 중 하나

4-2. 문맥 내에서의 few-shot learning

	NSMC (Acc)	KorQuAD (EA / 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 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 KLUE-YNAT and KLUE-STS from Park et al. (2021).

▲ 여섯 가지 작업에서의 few-shot 학습 결과
모델 크기가 증가함에 따라 다양한 문맥 내 학습 작업의 성능이 단조로이 향상됨
`Ko→ En` 번역 및 `KLUE-STS`의 경우 문맥 내 학습 능력이 베이스라인에 비해 훨씬 낮음

- 39B와 82B와 같은 **중간 크기 파라미터**를 가진 HyperCLOVA의 성능을 탐구
- 더 정교한 프롬프트 엔지니어링이 결과를 개선할 수 있을 것으로 기대됨

4-3. Prompt-based Tuning(P-tuning)

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 prompt-based 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 4000개 예제	90.7
w/ MLP-encoder	90.3
39B w/ p-tuning	**93.0**

성능이 비슷함

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.

- **p-tuning**은 메인 모델의 파라미터 업데이트 없이 비교 대상을 능가하도록 HyperCLOVA를 활성화
  - p-tuning이 HyperCLOVA의 강건성 뿐만 아니라 **정확도**를 크게 향상시키는 것으로 보임
- 내부 쿼리 수정 작업을 통한 실험(= 생성 작업에 대한 실험)
  - p-tuning을 통해 HyperCLOVA는 zero-shot 및 3-shot시나리오 모두에서 입력 쿼리 품질을 일관되게 **크게 향상**시킬 수 있었음

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**

⇒ LM의 규모가 증가함에 따라 이산 프롬프트를 전혀 사용하지 않아도 경쟁력 있는 성능 도출이 가능함

⇒ `GPT-3` 규모 모델의 입력 데이터에 대한 back gradient에 액세스 할 수 있는 경우 프롬프트 최적화 방법이 대규모 GPU 클러스터 없이도 NLP 연구자와 실무자들에게 대규모 LM의 표현 능력을 향상시킬 수 있는 대안임을 시사

4-4. 토큰화(Tokenization)의 영향

- 한국어 언어 특성을 고려
  → 형태소 인식 바이트 수준의 **BPE 토크나이제이션 방법**이 `KorQuAD` 및 두 가지 `AI Hub` 번역 작업에 미치는 영향을 분석함
- **Baseline**
  - 바이트 수준 BPE와 문자 수준 BPE를 사용
    - 영어 중심 말뭉치를 사용하여 사전 교육 LM을 토큰화하는 방법
  - 문자 수준 BPE = 원래 BPE
    - OOV(Out-of-Vocabulary)를 생성
    - '젝'과 같은 일부 한국 문자는 문자 수준 BPE 토큰에 포함되지 x
  - 다른 두 가지 토큰화 전략: OOV 토큰을 생성 x
  - 비교적 작은 크기인 13억 파라미터 모델을 사용

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

대부분은 성능 향상
훨씬 낮은 성능
성능 저하

- **HyperCLOVA**는 대부분의 작업에서 베이스라인과 비교하여 성능을 향상
  - Ko→En 작업에서는 형태소 분석기가 성능을 저하
- `YNAT`에서 문자 수준 BPE는 바이트 수준 BPE에 비해 훨씬 낮은 성능을 나타냄
  - 문자 수준 BPE가 일부 OOV 토큰을 생성하고 YNAT 데이터의 헤드라인에 포함된 일부 중요한 단어를 이해하기 어렵게 만들기 때문
  - ex) '프로젝트'라는 단어에 포함된 문자 '젝' (jec)은 문자 수준 BPE에서는 OOV 토큰이므로 '프로젝트'를 포함한 테스트 헤드라인을 이해하기 어려워짐

⇒ 영어와 언어적 특성 측면에서 매우 다른 언어에 대한 대규모 LM을 훈련시키기 위해 **언어별 토큰화**를 주의 깊게 디자인하는 것이 중요함

5. 대규모 언어 모델이 가져올 수 있는 산업적 변화

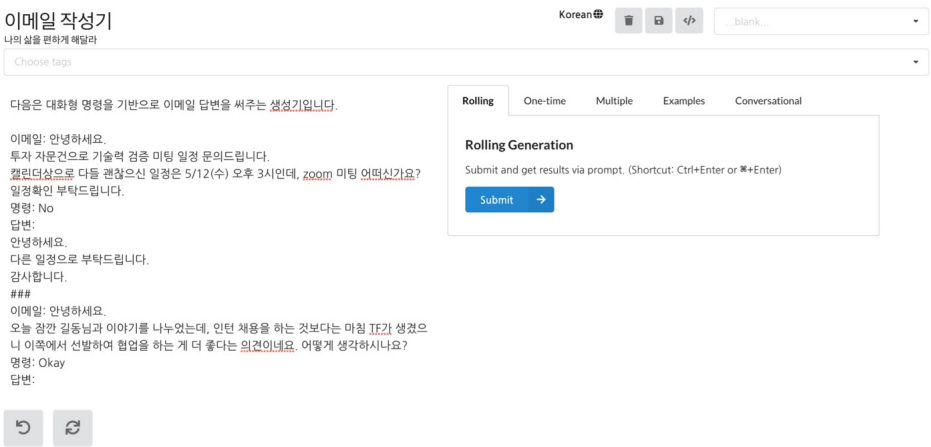
- **NLP ML 작업의 수명주기 가속화**
  - 대부분의 딥러닝 연구에서는 모델이 ML 전문가들에 의해 잘 수집된 데이터 세트로 훈련되고 그에 상응하는 명확하게 정의된 목표 함수와 함께 훈련되는 프로토콜을 따름
  - 제품 수준의 파이프라인에서 AI 제품을 만들기 위해선 몇 가지 추가 단계가 필요하며, 이는 엄청난 커뮤니케이션 오버헤드와 비용을 초래
  - 대규모 LMs 플랫폼을 사용하면 서비스 디자이너와 같은 개발자가 아닌 한 명만으로도 프로토타입 시스템을 구축할 수 있을 것으로 기대

5-1. HyperCLOVA Studio

- HyperCLOVA의 배포 방법

  → HyperCLOVA가 생성한 공유 아티팩트를 구축하고 공유하는 곳

- 두 가지 기능
  1. OpenAI Playground과 유사한 **GUI 인터페이스**를 제공



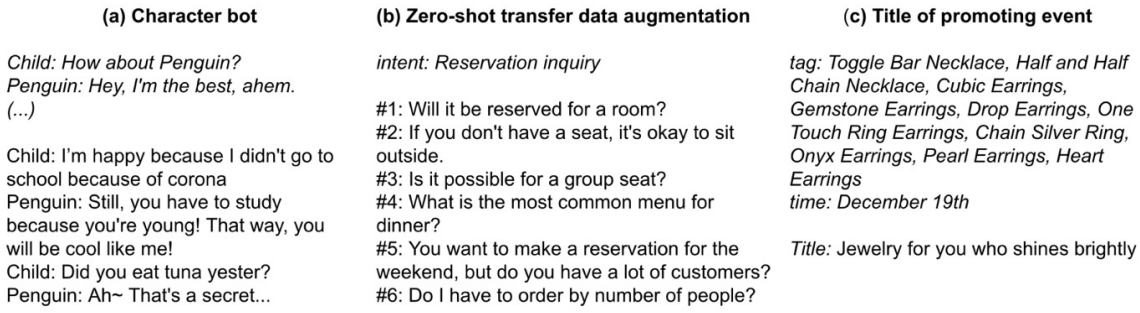
  2. 다양한 기능을 갖춘 **API end-point**를 지원

- ML 엔지니어의 참여를 최소화하면서 AI 기반 서비스의 신속한 프로토타이핑을 허용

5-2. HyperCLOVA Studio 활용 사례

- 내부 사용 사례는 아래의 세 가지 특성을 공유

1. 자동으로 목표 함수를 정의하거나 모델을 평가하는 것이 간단하지 않음
2. 입력과 출력의 스타일을 쉽게 제어할 수 있음
3. 프로그래밍 기술이나 AI 지식이 없는 제품 디자이너가 몇 시간 내에 쉽게 PoC(Concept of Proof) 시스템을 만들 수 있음



- **(a) 캐릭터 퍼스널리티를 가진 챗봇의 프로토타입을 빠르게 구축하기**
  - `HyperCLOVA`를 사용하여 캐릭터의 특성에 대한 한 두 줄의 설명과 몇 가지 대화 예제를 사용하여 특정 캐릭터 퍼스널리티를 가진 챗봇을 만들 수 있음

    → 메타버스 애플리케이션에서 여러 봇을 생성하는 데 사용할 수 있음
  - 프롬프트에서 대화 예제 몇 개를 변경하여 캐릭터의 스타일을 쉽게 제어할 수 있음
  - 프롬프트의 시작 부분을 사용하여 HyperCLOVA의 지식을 암묵적으로 추출할 수 있음

    ex) 유명인의 지식
  - Concept of Proof (PoC)를 쉽게 사용 가능
    - 인간 중심 프로세스를 통해 봇을 빠르게 만들 수 있음
    - 다양한 특성을 가진 대화 시스템을 빠르게 구축할 수 있음
- **(b) zero-shot 전이 데이터 증강**
  - 사용자의 의도에 맞게 조정된 발언을 생성하는 작업

    → 사용자 의도의 자연어 이름을 제공하면 해당하는 발언이 생성
  - 소스 도메인 클래스와 각 소스 도메인 클래스에 해당하는 예제를 프롬프트에 제공
  - 대화 시스템의 유효성을 검사하기 위해 다양한 발언을 생성
    - HyperCLOVA Studio를 사용하여 복잡한 상황/의도의 다양한 발언을 쉽게 생성할 수 있음
  - 실험 설계

- 내부 의도 말뭉치에서 대상 도메인으로 20개의 클래스를 선택하고 소스 도메인으로 6개의 클래스를 선택하고 각 클래스당 5개의 예제를 선택함

Zero-shot (Acc)	# of augmented samples (k)				
n	5(1)	10(2)	15(3)	25(5)	125(30)
0(0)	$60.8_{9.3}$	$68.9_{4.0}$	$71.9_{2.7}$	$74.8_{2.5}$	$78.0_{2.3}$

Few-shot (Acc)	# of original samples (n)				
k	1(1)	2(1)	3(1)	4(1)	5(1)
0(0)	$26.8_{6.0}$	$52.0_{4.9}$	$64.7_{5.2}$	$76.5_{4.4}$	$83.0_{3.0}$
25(5)	$79.2_{2.5}$	$81.2_{2.5}$	$82.6_{2.6}$	$83.4_{1.9}$	$84.3_{2.0}$
125(30)	$80.7_{2.2}$	$82.7_{1.9}$	$83.7_{2.1}$	$86.3_{1.5}$	$87.2_{1.7}$

- **(c) 이벤트 제목 생성**
  - 제품 광고를 향상시키기 위한 이벤트 제목을 생성하는 작업
  - 제품 특성을 설명하는 키워드를 인상적인 이벤트 제목으로 변환하는 Seq-to-Seq 작업
  - HyperCLOVA는 GT와 비교할 만한 고품질 제목을 생성함

	BLEU	Win	Lose	Tie
mT5 vs. GT	13.28	0.311	**0.433**	0.256
HyperCLOVA vs. mT5	-	**0.456**	0.350	0.194
GT vs. HyperCLOVA	5.66	0.311	0.333	**0.356**

Table 8: Results of event title generation. GT denotes the ground truth title written by human experts. *Win* means *X* wins against *Y* under *X* vs. *Y*. BLEU is the BLEU score of each model with its corresponding GT.

  - HyperCLOVA는 프롬프트를 수정하여 다른 도메인의 이벤트에 쉽게 적용할 수 있음
    - 동일한 키워드에 대해 각 디자이너가 강조하고자 하는 주제를 쉽게 제어할 수 있음
    - 이벤트 제목 생성 작업과 유사한 프롬프트를 사용하여 실제 서비스에 적합한 경우의 99%를 달성하였다는 연구 결과 존재

5-3. HyperCLOVA Studio의 가능성

1. HyperCLOVA의 입력 그래디언트 API는 로컬 하향식 작업의 성능을 향상시키는 데 적용할 수 있음
    - 프롬프트 기반 최적화로 성능 향상
    - Autoprompt, p-tuning, 또는 prompt tuning과 같은 프롬프트 최적화 방법을 사용하여 자체 프롬프트 인코더를 훈련시킬 수 있음
2. 프롬프트 인젝션 모듈을 적용할 수 있음
    - 검색기에 의해 검색된 적절한 문서를 사용하여 HyperCLOVA를 위한 오픈 도메인 QA 리더로 사용할 수 있음
    - 지식이나 유사한 예제를 검색하여 HyperCLOVA의 성능을 향상시킬 수 있음
3. 입력 및 출력 필터는 HyperCLOVA의 오용을 방지하는 데 도움이 됨
    - OpenAI API 활용 → 민감하거나 윤리적으로 부적절한 문장을 생성 모니터링하기 위한 필터 제공

5-4. No/Low Code AI 패러다임

- **기존의 머신 러닝 개발 파이프라인**은 여러 단계를 거침
    - 하나의 단계에서 발생하는 문제가 다른 단계로 전파되며, 모델 배포 이후에도 반복적인 프로세스로서 단계를 재방문하고 업데이트해야 함
    - 주로 파이프라인에 대한 다양한 전문 지식과 다른 역할을 필요로 하며, 전문가 간의 커뮤니케이션을 용이하게 하는 데 기여하는 공유된 기초적인 아티팩트가 없기 때문에 번거로우며 리소스가 많이 드는 작업임
- **HyperCLOVA Studio**와 같이 프롬프트를 사용하는 GUI 인터페이스를 가진 단일 대규모 LM은 이러한 문제를 완화할 수 있음
    - 여러 단계를 하나의 단계로 통합할 수 있음
    - 통합된 단계에서는 예제 선별, 프롬프트 디자인, API 매개 변수 조정 및 API 통합이 동시에 진행될 수 있음
    - 하나의 대규모 LM을 사용한 접근 방식은 전문가 간의 커뮤니케이션 비용을 현저하게 줄일 수 있음
- **No-Code AI** 접근 방식은 Proof of Concept을 빠르게 반복하는 데 유용하거나 서비스를 대규모 모델의 순수한 생성 능력으로만 구축할 수 있는 경우 강력함

6. 결론

- **HyperCLOVA**라는 다양한 양상의 수십억 개 규모의 한국 중심 언어 모델을 소개함
  - 82억 개의 파라미터
  - 문맥 내 zero-shot 및 few-shot 성능에서 최첨단을 보여주며 프롬프트 기반 학습 방법을 통해 더욱 향상될 수 있음
  - HyperCLOVA Studio를 통해 모델을 공유할 것이며, 여기서 개발자가 아닌 사람들도 쉽게 자신의 AI 지원 제품을 만들 수 있음
    - No Code AI 패러다임을 가능하게 함