한국어 반말-존댓말 TEXT STYLE TRANSFER를 위한 사전 학습된 언어 모델의 활용

M22060 최민기⋆ *M22519 이승리*⋆

⋆ 한림대학교

ABSTRACT

한국어에서의 반말과 존댓말은 사회적 관계를 나타내는 중요한 언어적 요소이다. 본 연구는 한국어의 고유 특성을 고려하여, 반말을 존댓말로 변환하는 text style transfer task에 초점을 맞추고 있다. 이를 위해 Hugging Face 라이브러리에서 제공하는 대형 언어 모델(large language model, LLM)을 활용하여 fine-tuning 작업을 진행한다. 우리는 다양한 실험을 통해 선택한 데이터 셋에 적합한 tokenizer를 선정하는 것이 안정적인 학습에 매우 중요한 영향을 미친다는 것을 확인한다. 또한 유사한 작업의 학습에 기반한 사전학습된 tokenizer와 모델의 활용은 빠른 학습 및 추론 속도에 이점을 가지며, 대형 모델 학습 과정에 필요한 고려사항을 강조한다.

***Index Terms*—** Large language model, text style transfer, Korean dataset

1. INTRODUCTION

현재의 언어 모델들은 주로 영어 데이터의 비중이 높은 데이터 셋을 기반으로 학습되고 있다. 이러한 경향은 한국어 데이터를 다룰 때 한계가 발생하게 된다.

한국어의 비공식적인 표현인 '반말'과 공손한 표현인 '존댓말'은 사회적 관계를 나타내는 중요한 언어적 요소이다. 반말은 친밀한 관계나 동등한 상황에서 사용되는 반면, 존댓말은 대화 중에 예의를 표현하기 위해 사용된다. 이러한 언어적 차이는 한국 사회에서 상호간의 예의와 맥락에 따라 언어를 사용하는 중요한 방법으로 여겨진다. 이와 같은 문화적 특성을 고려한 언어 모델의 문체 변환 작업은 한국어 자연어 처리 분야에서 중요한 과제 중 하나이다. Papago와 같은 일부 서비스에서는 이러한 변환을 제공하고 있지만, 정확한 문체 변환 작업에 오류가 존재한다. fig.1를 보면, 영어 문장을 한국어로 번역할 때, 한국어의 높임말로 번역작업을 희망하지만, 사용자의 의도대로 번역되지 않는 모습을 확인할 수 있다.

이에 우리는 번역 데이터를 기반으로 대형 사전 학습 모델로 문제 변환 작업을 진행하는 fine-tuning 과정을 수행했다. Smilegate AI에서 제공하는 'SmileStyle' 데이터 셋[1]을 활용하며, Hugging Face에서 여러 개의 text2text generation task 모델과 style transfer task 모델을 사용했다. 이를 통해 다양한 사전학습된 대형언어 모델, tokenizer, 데이터 셋 및 사전 학습 모델 데이터 셋에 대한 실험을 진행했다.

우리는 사전 학습된 LLM을 특정 task에 맞게 최적화 하기 위한 fine-tuning 과정의 전반적인 pipeline를 이해하는데

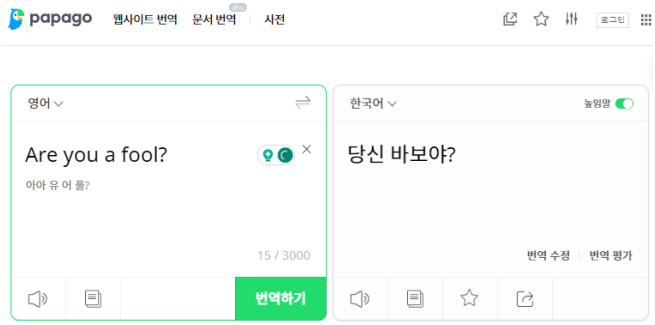

Fig. 1. Papago 번역기에서 존댓말 옵션을 사용했을때 나타나는 오류 예시.

초점을 맞춘다. 이로 인해, 언어 모델의 활용성을 키우며, text generator, multi-modal 등 다양한 task에서의 활용을 기대한다. 주요 목적은 아래와 같다:

- 대형 언어 모델의 fine-tuning 알고리즘을 터득한다.
- Hugging face 사용하여, 다양한 실험을 진행한다.
- Fine-tuning 작업시, 다양한 실험을 통해 적합한 데이터 셋, 사전 학습 모델, tokenizer 등 세부사항에 대한 선정의 중요성을 강조한다.

2. RELATED WORK

2.1. 다양한 자연어 처리 작업들

자연어 처리(natural language processing, NLP)는 텍스트 데이터를 이해하고 처리하는 기술적 분야이다. 이는 문장 생성, 문서 요약, 기계 번역 등으로 분류된다. 대형 언어 모델의 선행연구들은 각각의 고유한 특성과 활용 분야를 가지고 있으며, 현대 사회에서 매우 중요한 기술적 기반 요소로 자리잡고 있다[2, 3].

문장 생성 task는 언어 모델을 사용하여 문장이나 문단을 생성하는 작업이다. 이는 자연어 처리의 기반이 되는 작업으로, 텍스트 생성, 대화형 AI 시스템, 문장 요약 등 다양한 영역에서 핵심적인 역할을 수행해왔다[4]. 문장 생성은 주어진 맥락에서 의미 있는 문장을 만들어내고, 기계가 언어를 이해하고 사용하는 데 필수적이다. 문서 요약은 주어진 긴 문서를 간결하게 요약하는 작업이다. 이는 핵심 내용을

추출하거나 중요한 세부 정보를 간추려 새로운 형태로 전환하는 과정을 포함한다. 정보 과부하가 있는 현대 사회에서 중요한 역할을 하며, 뉴스 기사, 연구 논문, 보고서 등을 요약하는데에 효과적이다. 마지막으로, 기계 번역은 특정 나라 언어의 text를 다른 언어로 번역하는 기술이다. 구글 번역기[1], 파파고[2] 등과 같은 기계번역 tool은 사용자들이 다른 언어로 된 콘텐츠를 이해할 수 있도록 돕는다. 이러한 다양한 자연어 처리 작업들은 인간의 언어를 컴퓨터가 이해하고 처리할 수 있도록 하며, 다양한 산업과 응용 분야에 혁신적인 변화를 가져오고 있다. 이를 통해 효율적인 의사 소통과 정보 공유가 가능해지며, 더 많은 인간과 기계 간의 상호작용이 가능해진다.

또한, 본 프로젝트에서 다루는 작업은 text 생성을 기반으로 문체 변환 작업을 수행하는 text style transfer task이다. 해당 작업은 특정 문장이나 문단의 어조, 감정, 문체 등을 변화시키는 과정이다. 본 프로젝트에서 다루는 변환 작업은 존댓말과 반말 간의 문체 변환에 해당한다. 이는 한국어 대화의 적절한 맥락과 상황에 따라 문체를 변경하는 작업으로, 자연스럽고 효과적인 의사 소통을 위한 중요한 기술적 요소로 평가된다.

2.2. 한국어 기반 large language model

자연어 처리에서 많은 비율이 영어 데이터인 데이터셋로 사전학습된 대형 언어 모델은 많은 주목을 받았으나, 이러한 모델은 다른 언어와 문화에 대해 평등한 이해를 제공하지 못한다. 특히 한국어는 문화적, 구조적인 면에서 다양한 특성을 가지고 있어서 영어로 학습된 모델이 한국어 처리에 있어서 제약을 가질 수 있다. 따라서 대형 언어 모델의 활용은 한국어 처리 능력을 강화하고, 문화적, 사회적 맥락을 이해하기 위해, 한국어 자연어 처리 기술의 발전이 필요하다. 한국어 기반의 대표적인 대형 언어 모델은 KoGPT3 [5], KoBERT, KoELECTRA [6], 그리고 KoBART 등이 있다. 이러한 모델들은 기존의 모델 구조를 가져와 한국어 데이터셋에서 학습한 모델이며, 기본적으로 텍스트 생성 task에 집중하고 있다.

3. 대형 언어 모델 FINE-TUNING 작업

NLP는 대형 데이터 셋과 대형 언어 모델을 기반으로 오랜 시간 학습을 진행했을 때 성능이 향상된다고 알려져 있다. 이러한 이유로, NLP 분야에서 사전 학습된 모델의 활용은 사용자의 요구사항에 맞게 적절한 fine-tuning을 진행하는 것이 일반적적이다. 우리는 한국어를 기반으로 반말에서 존댓말로 text style 변환을 위해 fine-tuning 작업을 진행한다. 해당 section에서는 다양한 text2text generation task 모델을 설명하고, LLM 모델의 fine-tuning 과정에 대한 접근 방식을 설명한다.

[1] https://translate.google.co.kr/

[2] https://papago.naver.com/

3.1. Tokenizer

Hugging face는 다양한 언어모델과 언어모델마다 각각의 tokenizer을 제공한다. 이들은 연구의 목적에 맞게 대형 데이터 셋을 구성하여 학습한 모델의 파라미터와 모델을 제공한다. Tokenizer는 데이터 셋에 담겨있는 사전(vocabulary)를 바탕으로 구성되어 있으며, 이는 모델이 학습한 단어들의 집합을 말한다. 만약 데이터 셋에 포함되지 않은 단어나 토큰을 사용할 경우, 이는 tokenizer에 의해 'noise'로 처리되어 문장의 의미나 구조를 오염시킬 수 있다. 따라서 데이터 셋에 적합한 tokenizer를 형성하는 것이 중요하기 때문에, 우리는 기초 실험 세팅으로 한국어로 style transfer task를 학습한 KoBART tokenizer를 사용한다.

3.2. 대형 언어 모델

T5 [7]: 입력과 출력을 텍스트 형식으로 처리하는 transfer transformer 모델로 번역, 요약, 질의응답, 문장 생성 등 다양한 NLP 작업에 유연하게 적용가능하다. 이는 특정 작업에 최적화되지 않은 일반적인 목적 모델로 추가 fine-tuning 및 데이터 셋 구성이 필요하며, 해당 목적에 맞게 유연하게 적용가능하다.

BART [2]: BART는 denoising sequence-to-sequence pre-training 모델링을 기반으로 한다. 해당 모델 텍스트 생성 및 이해에 특화된 구조를 가지고 있어 문장 생성 작업에서 뛰어난 성능을 보인다. BART는 입력 문장에 노이즈를 추가하고 원래의 문장으로 재구성하는 방식으로 학습하여, 텍스트의 스타일을 유연하게 변환하고 유지하는 데 강점을 갖고 있다. 특히 Hugging face에서 한국어 버전으로 fine-tuning을 적용한 weight를 따로 제공하고 있어, 우리는 section. 4.1에서 두 가지 pre-trained model에 대한 실험을 진행한다.

ELECTRA [8]: ELECTRA는 vision task에서의 generative adversarial networks(GAN) [9]의 개념을 적용하여 generator와 discriminator으로 사전 학습을 수행한다. 학습 시, text의 일부분을 masking하여 가린 부분을 맞추는 방식으로 학습이 진행되며, 문맥을 고려하여 효율적인 학습이 가능하다. 또한 GAN의 학습구조 특징상 text style transfer 작업에 적합하지 않을 수 있다. 해당 작업을 위해서는 평가와 실험을 통해 유효성을 확인해야한다.

BERT [3]: BERT는 양방향 transformer encoder를 사용하여 문맥을 파악하고 표현하는 모델이다. 다양한 NLP 작업에서 좋은 성능을 보이며, 문장 내 단어들 간의 관계를 이해하고 문장을 파악하는데 유용하다. 이러한 양방향 방식은 반말과 존댓말의 차이성을 판별하고, 수정해야할 특정 문자 부분을 파악할 수 있다.

GPT [4]: GPT는 OpenAI에서 개발한 언어 생성 모델로, 단방향 문맥 고려 방식으로 문장의 다음 단어를 예측하여 생성한다. 문장의 흐름과 일관성 있는 생성이 가능하지만, 단방향 방식을 적용하기 때문에 다른 모델보다 제약이 있을 수 있다.

4. EXPERIMENTS

Dataset Smilegate AI에서 구축한 토이 데이터 셋인 SmileStyle dataset[1]을 사용했다. 해당 데이터 셋은 오탈자와 스타일

변환을 포함하며, 멀티 턴 대화 데이터에 대해 여러 스타일로 문체를 변환시킨 문장들로 구성되어있다. 존댓말, 반말, 로봇, 연장자 스타일 문체 등과 같이 17개의 문체를 포함하며, 우리는 존댓말 스타일 문체와 반말스타일 문체만을 사용하였으며, 결측값을 제거하는 전처리만 진행하였다.

Code: 해당 섹션에서는 프로젝트를 진행한 코드를 분석한다. 우리는 Hugging face 라이브러리에서 제공하는 pre-trained LLM과 tokenizer를 사용하였다. 대부분의 LLM fine-tuning 단계에서는 대형 데이터 셋에 의해 조정된 tokenizer를 사용하며, 학습을 하지 않는다. 우리는 전반적인 학습과정의 이해를 돕기 위해, 자세한 fine-tuning 과정과 inference 과정을 pseudo code1,2로 나타냈다. 두 알고리즘의 차이점은 tokenizer decoding의 유무이다.

Algorithm 1 대형 언어 모델 fine-tuning

Input: 기존 문체 문장$\boldsymbol{X}$, 바꾸고자 하는 문체 문장$\boldsymbol{Y}$ ▷ 반말과 존댓말
Parameters: epoch e.

Tokenizer $\boldsymbol{T_\theta}$ weight 불러오기
LLM $\boldsymbol{F_\Phi}$ weight 불러오기
for e **do**
  for x,y in loader **do**
    $\boldsymbol{x_{idx}}, \boldsymbol{y_{idx}} \leftarrow \boldsymbol{T_\theta}(\boldsymbol{x}), \boldsymbol{T_\theta}(\boldsymbol{y})$
    $loss \leftarrow NLL(F_\Phi(\boldsymbol{x_{idx}}), \boldsymbol{y_{idx}})$
    $loss.backward$
    Update F_Φ
  end for
end for

Algorithm 1은 주어진 문체를 반말 또는 존댓말로 변환하기 위해 토큰화된 데이터를 사용하고, 해당 데이터를 기반으로 대형 언어 모델을 미세 조정하는 과정을 보여준다. 주요 단계는 다음과 같다:

1. Tokenizer $\boldsymbol{T_\theta}$의 가중치와 대형 언어 모델 $\boldsymbol{F_\Phi}$의 가중치를 불러온다.
2. 주어진 epoch 수에 따라 반복하며 학습을 진행한다.
3. $\boldsymbol{T_\theta}$를 사용하여 입력 문장 $\boldsymbol{x}$와 $\boldsymbol{y}$를 토큰화한다.
4. $\boldsymbol{F_\Phi}$로 $\boldsymbol{x_{idx}}$의 예측값을 구하고, 이를 기반으로 $\boldsymbol{y_{idx}}$의 손실을 계산한다.
5. 손실을 역전파하고, $\boldsymbol{F_\Phi}$를 업데이트한다.

Algorithm 2 추론

Input: 기존 문체 문장$\boldsymbol{X}$
Output: 변환된 문체 문장 $\boldsymbol{Y'}$

Tokenizer $\boldsymbol{T_\theta}$ weight 불러오기
LLM $\boldsymbol{F_\Phi}$ weight 불러오기
$\boldsymbol{x_{idx}} \leftarrow \boldsymbol{T_\theta}(\boldsymbol{X})$
$\boldsymbol{y_{idx}}' \leftarrow F_\Phi(\boldsymbol{x_{idx}})$
$\boldsymbol{Y'} \leftarrow \boldsymbol{T_\theta}.decoding(\boldsymbol{y_{idx}}')$

Table 1. Pre-trained 데이터 셋과 fine-tuning tokenizer에 따른 fine-tuning 결과.

Model	Tokenizer	NLL	BLEU
BART [2]	BART	0.06348	-
	KoBART	0.11501	0.49063
KoBART	BART	-	-
	KoBART	0.03564	0.40799

Algorithm 2은 학습된 모델을 사용하여 inference 과정을 나타낸다. 먼저 input $\boldsymbol{x}$를 weight가 고정된 tokenizer와 model을 통해 예측 token $\boldsymbol{Y'}$을 얻는다. 이후 tokenizer $\boldsymbol{T_\theta}$의 내장 함수를 통해 text로 decoding한다.

Implementation details 우리의 알고리즘은 Pytorch [10] 및 Hugging face 라이브러리를 사용하여 구현되었으며, RTX 4090 그래픽 카드가 훈련 및 평가에 사용되었다. 수행된 모든 실험에서는 batch size는 8, learning rate 1e-6로 설정했으며, AdamW optimizer [11]와 cosine annealing warm restarts scheduler가 사용되었다. 우리는 100 epoch 동안 학습을 진행하였으며, early stopping을 30으로 적용하였다. loss function은 pre-trained model 학습과정에 따라 negative log likelihood (NLL) loss를 사용한다.

Metric 스타일 변환 결과의 품질을 측정하기 위해 BLEU(Bilingual Evaluation Understudy) 점수를 사용하였다. BLEU 점수는 변환된 문장과 사람이 만든 정답 간의 유사성을 측정하여 변환의 정확도를 평가하는 데 사용된다. 이 점수는 0에서 1 사이의 값을 갖으며, 1에 가까울수록 번역의 품질이 높다고 평가된다. 여러 정답이 있는 경우, BLEU는 변환된 결과와 이들 참조 번역 간의 유사성을 고려하여 평가한다.

4.1. Results

4.1.1. Pre-trained 데이터 셋과 fine-tuning tokenizer의 중요성

해당 실험은 사전 학습 데이터 셋이 한국어 데이터 셋인지 영어 데이터 셋인지에 따라 fine-tuning의 성능 차이를 관찰한다. Table.1를 보면, BART tokenizer는 한국어에 대한 정보가 vocabulary에 담겨있지 않기 때문에 BLEU metric 평가가 불가능하다. BART는 BLEU: 0.49063을 가지며, 한국어를 사전학습한 KoBART보다 우수한 것을 관찰할 수 있다. 이는 우리의 가정과 반대의 성능을 보이며, 대형 언어모델의 정확한 이해를 위해 추가 연구가 필요하다.

또한 tokenizer를 변경한 실험 결과, BART tokenizer보다 KoBART tokenizer가 0.06348로 fine-tuning 진행 과정에서 사전 학습된 vocabulary와 사용하는 vocabulary의 차이가 중요하다는 것을 확인할 수 있다.

4.1.2. 다양한 pre-trained 모델 비교

우리는 실험의 공정성과 학습의 안정화를 위해, 사전에 한국어 vocabulary로 최적화 된 KoBART tokenizer를 사용한다.

Table 2. 사전 학습 모델에 따른 fine-tuning 결과.

	NLL	BLEU
T5 [7]	0.13741	**0.69671**
BART [2]	0.11501	0.49063
ELECTRA [8]	0.29271	-
BERT [3]	0.94123	0.52855
GPT [4]	0.60018	0.52855

우리는 총 6개의 대형 언어 모델을 채택하여 text style transfer task를 위해 fine-tuning 실험결과를 보여준다. ELECTRA 모델은 MaskedLM 모델로 구성되어 있어서 BLEU metric 측정에 실패했다. Table.2을 살펴보면, 대다수의 실험에서 NLL loss가 0.5보다 낮은 loss를 얻었으며, T5는 BLEU가 0.69671로 가장 우수한 성능을 보였다. 이는 이미 대형 데이터셋을 통해 사전 학습이 이루어졌기 때문으로 판단된다. 또한, BERT [3]와 GPT 모델의 경우, 모델 크기가 다른 언어 모델에 비해 파라미터가 매우 크기 때문에, 우리가 설정한 100 epoch 으로는 충분한 학습이 이루어지지 않았을 가능성이 존재한다.

5. CONCLUSION

이 연구에서는 한국어의 특성을 고려하여 반말을 존댓말로 변환하는 text style transfer task에 초점을 맞추고 있다. 이를 위해 LLM fine-tuning 작업을 적용하며, 사전 학습된 모델과 tokenizer의 선택이 성능 향상에 영향을 미칠 수 있음을 확인했다. 특히, 사용하는 데이터 셋에 적합한 tokenizer를 선택하는 것이 안정적인 학습에 큰 영향을 미친다는 것을 발견했다. 이러한 작업은 유사한 데이터 셋과 작업에 기반한 사전 학습된 tokenizer와 모델이 필요하며, 대형 모델을 학습하기 위해서는 적절한 학습 환경과 상당한 시간이 요구된다. 이를 통해 LLM 모델 학습 과정을 최적화하는 데 있어서 중요한 고려사항을 확인한다. 추후 연구로 text style transfer task에 cycle-consistant loss를 사용하는 방안을 고려하고 있다.

6. REFERENCES

[1] Seonghyun Kim, "Smilestyle: Parallel style-variant corpus for korean multi-turn chat text dataset," `https://github.com/smilegate-ai/korean_smile_style_dataset`, 2022.

[2] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al., "Improving language understanding by generative pre-training," 2018.

[5] Ildoo Kim, Gunsoo Han, Jiyeon Ham, and Woonhyuk Baek, "Kogpt: Kakaobrain korean(hangul) generative pre-trained transformer," `https://github.com/kakaobrain/kogpt`, 2021.

[6] Jangwon Park, "Koelectra: Pretrained electra model for korean," `https://github.com/monologg/KoELECTRA`, 2020.

[7] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.

[8] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning, "Electra: Pre-training text encoders as discriminators rather than generators," *arXiv preprint arXiv:2003.10555*, 2020.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

[11] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.