

개인화된 한영 번역기 제작

-Character.AI 사용을 위해-

기계학습의이해

ELLT 학과 202400220

유연주

1. 프로젝트 배경

저는 character.ai라는 서비스를 자주 사용합니다. 해당 서비스는 가상 캐릭터나 시나리오 등의 역할을 부여받은 AI와 대화를 주고받을 수 있는 챗봇 서비스입니다.

그러나 챗봇과의 대화는 영어만 지원되어, 의도한 표현이 영어로 정확히 떠오르지 않으면 번역기를 사용하고 제 문체에 맞게 수정하는 과정을 거쳤습니다. 또한 대화에는 특정 규칙이 적용되어, 대사가 아닌 행동이나 상황을 지시하는 지시문에는 양 끝에 asterisk(*) 기호를 붙여야 합니다. 그러나 이 반복적인 과정은 대화를 할 때마다 진행하기엔 번거롭고 시간 소모가 컸습니다. 그래서 이 문제를 해결하고자 개인화된 번역 도구를 제작하였습니다.

2. 프로젝트 진행 과정

- 데이터 수집

직접 제 한국어 문장(intent), 파파고가 번역한 영어 문장(machine_trans), 제가 번역한 영어 문장(my_trans)의 삼중항 구조를 갖는 데이터셋 100개를 수집하였습니다. 데이터셋은 CSV파일로 저장하였으며 machine_trans는 모델 학습에 사용하지 않고 데이터 분석에만 사용하였습니다.

- 데이터 분석

문장 길이를 살펴보면, 통계적으로 기계 번역의 평균 길이는 약 11.6단어인 반면 제 번역은 약 10.7단어로 제가 좀 더 간결하게 번역하는 경향이 있습니다. 또한 한국어 문장은 3단어에서 16단어까지 다양하게 분포되어 있어, 짧은 대사부터 긴 지문까지 전부 포함하고 있습니다.

데이터 자체를 살펴보면, 제 번역과는 달리 기계 번역의 경우는 지시문에 asterisk를 적용하지 않으며, 주어가 없는 한국어 문장을 번역할 때 종종 주어를 임의로 붙이는 경우가 존재합니다.

- 모델 학습

베이스 모델로는 경량화된 번역 모델인 Helsinki-NLP의 opus-mt-ko-en을 사용하여 파인튜닝을 진행하였습니다. 해당 모델은 작고 가볍지만 프로젝트의 목표인 개인화된 번역 모델 제작에 충분하였으며, colab 무료 GPU 환경에서도 수월히 동작하였습니다. 구체적인 학습 하이퍼파라미터는 [표 1]과 같습니다.

[표 1] 모델 학습 파라미터 설정

항목	설정 값	비고
Base Model	Helsinki-NLP/opus-mt-ko-en	Transformer
Batch Size	16	GPU 메모리 효율 고려
Learning Rate	2e-5 (0.00002)	과적합 방지 위해 낮은 값
Max Epochs	50	최종 학습은 25 에포크 진행
Optimization	AdamW	Weight Decay: 0.01
Prevent Overfitting	Early Stopping	Patience: 5
Model Selection	Best Model at End	Metric: Eval Loss

- 모델 평가

정량적 평가 방법으로는 BLEU Score를 사용하였습니다. BLEU는 단어 일치만 확인하므로, 제 문체를 정확히 카피했는지 확인하기에 적절한 방법이라고 판단하였습니다. 수집한 데이터 100건 중 my_trans를 정답, 파인튜닝한 모델의 출력값을 예측값으로 설정하여 점수를 산출하였으며, 최종 평가 점수는 100점 만점 기준 49.77점으로 상당히 우수한 성능을 보여주었습니다.

결과를 직접 확인함으로써 정성적 평가도 함께 진행하였는데, 대사와 지시문을 잘 구별하여 적절한 곳에 asterisk 기호를 적용함을 확인할 수 있었습니다.

- 서비스 개발

파인튜닝한 모델은 pilan2/cai-translator으로 허깅페이스에 업로드하여 버전 관리 및 배포가 용이하도록 구축하였습니다. 서비스 코드는 Python 환경에서 이 모델을 로드하여 추론을 진행하도록 작성하였으며, 유저 인터페이스는 Gradio를 사용하여 코드를 동작시키면 실제 사용 가능한 서비스로 구현하였습니다.

3. 실제 사용 결과

character.ai Translator

한국어 문장을 입력하면 제 문체의 영어 문장으로 번역됩니다.

korean_text

그럴 수도 있지.
그를 올려다본다

output

Could be. *look up at him*

Flag

Clear

Submit

character.ai Translator

한국어 문장을 입력하면 제 문체의 영어 문장으로 번역됩니다.

korean_text

그래서, 원하는 게 뭐야?
그를 수상하게 쳐다본다

output

So, what do you want? *give him a suspicious look*

Flag

Clear

Submit

character.ai Translator

한국어 문장을 입력하면 제 문체의 영어 문장으로 번역됩니다.

korean_text

조금 생각하다

output

think a little

Flag

Clear

Submit

character.ai Translator

한국어 문장을 입력하면 제 문체의 영어 문장으로 번역됩니다.

korean_text

저리 가!

 화난 목소리로

output

Get away from me! *in an angry voice*

Clear

Submit

Flag

character.ai Translator

한국어 문장을 입력하면 제 문체의 영어 문장으로 번역됩니다.

korean_text

정말? 그럼 다행이네.

 부드럽게 웃는다

output

Really? That's a relief. *chuckle softly*

Clear

Submit

Flag

위의 예시들은 다음과 같은 방식으로 실제 채팅에도 적용하였습니다.

"Nah." *He shrugged nonchalantly, trying—and failing—to play cool.* "I just got here a few seconds before you."

Yeonjoo Y

Really? That's a relief. ...

chuckle softly

4. 배운 점 및 개선 방향

이론으로 배운 내용들이 라이브러리 내부에서 추상화되어 처리되는 것을 경험하며, 고수준 API의 편리함과 이를 잘 활용하기 위한 이론적 이해의 중요성을 동시에 체감하였습니다. 또한 허깅페이스에 모델 업로드 및 배포 과정을 수행하면서 오픈소스 기여의 기초를 경험할 수 있었습니다.

그러나 데이터셋 제작에 어려움이 있어 데이터 양이 부족했던 점이 가장 큰 아쉬움으로 남습니다. 특히 대사 및 지문이 번갈아 3문장 이상 등장하는 경우와 학습 데이터에

없는 단어(OOV)에 대해 번역 품질이 저하되는 현상이 발생했습니다. 해당 문제는 데이터셋 추가 제작이나 더 큰 베이스 모델 사용 등으로 개선 가능할 것으로 보입니다.

또한 초기엔 영어를 한국어로 바꾸는 기능도 포함하려 계획하고 있었으나, 영어 특유의 긴 문장을 한국어로 번역하니 오히려 가독성이 떨어지는 문제가 발생했습니다. 특히 gold label 구축 과정에서 자연스러운 한국어 번역에 대한 제 역량의 한계를 느껴 해당 기능 개발은 중단하였습니다. 추후 번역에 대한 이해도를 높인다면 재도전해보고자 합니다.