최종보고서

오픈소스SW기여

Project name: 향상된 번역기

Members: 김명현

Contents

1. Introduction & Motivation

한글을 영문으로 번역하는 과정에서 짧은 문장이나 단어는 정확하게 번역이 가능하다. 정확하게, 기존에 사람들이 많이 번역했던 텍스트는 많은 양의 데이터가 학습되어 있기 때문에 정확한 번역 품질을 보여주지만 이해와 관련하여 번역 오류가 발생한다. 예를 들어, 기존에 많이 번역되지 않아 축적된 데이터의 양이 적은 희귀언어의 경우, 번역물의 정확성이 매우 떨어진다는 한계를 지닌다. 이러한 경우는 새로운 창작물이나 논문을 영작할 때 구글 번역을 사용하는 경우이며, 주로 문단이나 긴 문장의 전체적인 흐름을 통한 이해가 수반되어야 한다.

따라서 이 프로젝트의 목표는 단순 기계 번역이 아닌 사전지식이 요구되는 어느정도 학문적인 내용의 글을 자연스럽게 번역하기 위해 Google 번역과 ChatGPT 오픈소스를 활용하여 기존보다 정확성이 높은 번역기를 만들고자 한다. 이로 인해 기대할 수 있는 효과는 긴 문장이나 하나의 문단을 원어민이 사용할 만한 표현을 사용하여 매끄럽게 번역을 할 수 있고 영어가 아닌 타 언어로 된 논문도 번역을 통해 읽을 수 있을 것이다.

2. Requirement

프로젝트에서 제공하고자 하는 기능은 다음과 같다.

<table>
<tr><th>기능</th><th>내용</th></tr>
<tr><td>자동 언어 탐지</td><td>사용자가 따로 입력한 글의 언어가 무엇인지 선택하지 않아도 입력된 텍스트의 언어를 탐지할 수 있다. 언어 선택창이 입력과 결과창 둘 다 있으면 선택 가능한 언어가 많기 때문에 찾는데 번거롭고 귀찮다. 따라서 자동 언어 탐지 기능이 있다면 사용자에게 편리함을 제공할 수 있다.

Format: translator.detect (lang, confidence)

<table>
<tr><th>요소</th><th>의미</th></tr>
<tr><td>lang</td><td>감지한 언어</td></tr>
<tr><td>confidence</td><td>결과의 신뢰도 (0~1.0)</td></tr>
</table>

감지 가능한 언어 리스트
LANGUAGES = {
'ko': 'korean',
'zh-cn': 'chinese (simplified)',
'nl': 'dutch',
'en': 'english',
'de': 'german',
'el': 'greek',
'hi': 'hindi',
'ga': 'irish',
'it': 'italian',
'ja': 'japanese',
'la': 'latin',
'ru': 'russian',
…
106 개 언어 지원
}</td></tr>
</table>

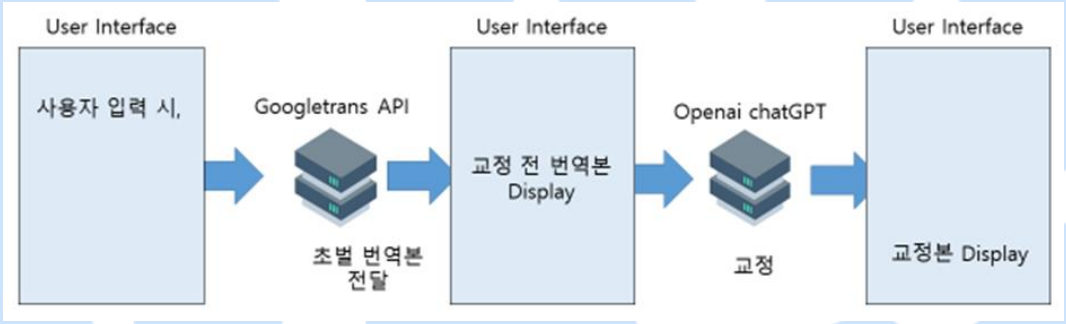
<table>
<tr><td>번역</td><td>내가 입력한 언어를 바꾸고자 하는 언어로 바꿔주는 기능이다. 단순이 번역 결과만을 알려주는 것 보다도 번역 결과의 발음까지 알 수 있는데, 이는 발음이 생소한 언어로 번역할 때 유용하게 사용할 수 있다. 번역 버튼을 누르면 번역+교정이 이루어져 최종 번역본이 결과창에 뜨게 된다.

Format: translator.translate(src, dest, text, pronunciation)
<table><tr><th>요소</th><th>의미</th></tr><tr><td>src</td><td>번역 전 언어</td></tr><tr><td>dest</td><td>번역 후 언어</td></tr><tr><td>text</td><td>번역 결과</td></tr><tr><td>pronunciation</td><td>번역 결과의 발음 (영어로 표기)</td></tr></table>Ex) KOR to ENG
Translator.translate('안녕', src='ko', dest='en').text
>>> 'Hi'

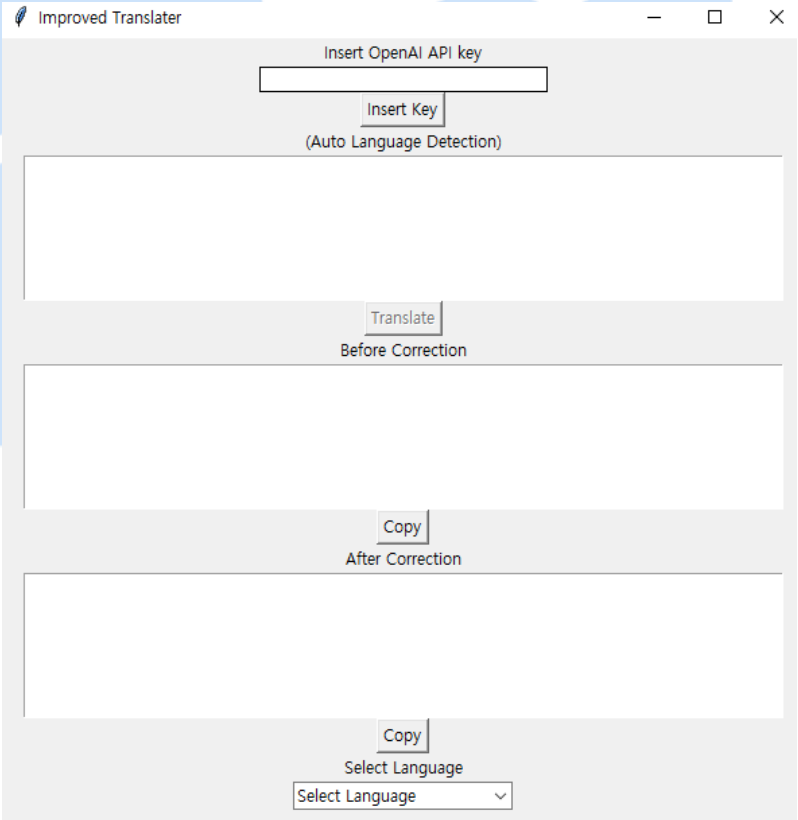
번역하고자 하는 언어 또한 지원하는 내에서는 자유롭게 선택하여 번역할 수 있다. 언어 선택 버튼을 누르면 번역 가능 리스트 중에 사용자가 원하는 언어를 선택하고 번역을 진행하면 된다.</td></tr>
<tr><td>교정</td><td>구글 번역에게서 받은 초벌 번역본을 ChatGPT를 통해 교정을 진행한다. 사용자가 따로 커스텀할 수는 없는 기능이지만 번역을 위해 선택된 언어로 교정이 다시 진행되며 이는 기계 번역의 한계를 보완해줄 수 있어 사용자에게 정확하고 세련된 작문을 제공할 수 있다.</td></tr>
<tr><td>클립보드</td><td>결과로 나온 최종 번역본을 ctrl + c, ctrl + v하지 않고도 결과창 옆에 있는 복사 버튼만 누른다면 최종 번역본이 자동으로 복사가 되어 필요한 곳에 ctrl + v만 하면 붙여넣기가 가능하다.</td></tr>
<tr><td>API키 입력</td><td>Chatgpt API를 사용하기 위해 필요한 사용자의 API 키를 삽입할 수 있는 기능이다. 키 입력창에 키를 입력하고 삽입 버튼을 누르면 키 입력이 완료된다.</td></tr>
</table>

번역할 언어 선택	Googletrans에서 제공하는 언어 106가지를 모두 제공 가능하다. 번역기 맨 아래에 있는 언어 선택 창을 통해 언어를 선택할 수 있다.
기타	잘못된 접근을 통한 버그를 막기 위해 기능들의 예외처리를 하여 완성도를 높였다.

3. Design



전체적인 구조 설계는 위 그림과 같다. UI에서 사용자가 번역할 텍스트를 입력하면 googletrans API를 이용해 번역한다. 이렇게 나온 번역본을 ChatGPT 모델에게 교정을 맡겨 교정본을 만든다. 이후 번역본과 교정본을 UI창이 출력하여 기능을 제공한다. 오른쪽은 User Interface를 캡쳐한 그림이다.



4. Developing

주요 기능들에 대한 코드이다. 번역과 교정은 각각 Googletrans와 Openai API를 사용하여 Python을 통해 구현하였다.

```
#번역
translator = Translator()
self.detected = translator.detect(x)  # 입력받은 문장의 언어 감지
self.src_lang = self.detected.lang
if x is not None:
  self.translated_text = translator.translate(x, src=self.src_lang, dest=self.dest_lang).text

# openai의 GPT 모델을 사용하여 문장을 교정합니다.
openai.api_key = self.key_val
model_engine = "text-davinci-003"
self.corrected_text = openai.Completion.create(
    engine=model_engine,
    prompt=(f"Please polish and correct following these sentences in {self.dest_lang}: {self.translated_text}"),
    temperature=0.5,
    max_tokens=2048,
    n=1,
    stop=None,
```

또한 예외처리를 통해 다음과 같은 기능을 제공하여 작품의 완성도를 높였다.

1) API 키 유효성 검사: key_insert() 함수에서 API 키를 입력하는 부분에서 유효성 검사를 추가하는 것이 좋다. 예를 들어, API 키가 올바르지 않거나 입력되지 않은 경우 사용자에게 알리고 처리를 중단할 수 있다.

```
if not key_val.strip():  # API 키가 입력되지 않은 경우 예외 처리
        messagebox.showerror("Key Error", "Please enter a valid API key.")
        return
```

2) 입력값 유효성 검사: 사용자가 입력한 문장이 없거나 비어 있는 경우에 대한 예외 처리를 추가하는 것이 좋다. 예를 들어, translate_text() 함수에서 x 값이 None이거나 빈 문자열인지 확인하고, 이에 따라 적절한 오류 처리를 수행한다. 또한 API의 번역 제공 한도를 맞추기 위해 입력받은 텍스트가 긴 경우에도 오류 처리를 수행한다.

```
if not x.strip():  # 입력값이 없거나 공백인 경우 예외 처리
        messagebox.showerror("Translation Error", "Please enter a sentence to translate.")
        return
if len(x) > 5000:  # 메시지가 5000자를 초과하는 경우 예외 처리
        messagebox.showerror("Translation Error", "Please enter a message with a maximum of 5000 characters.")
        return
```

3) 예외 메시지 출력: 발생한 예외에 대한 메시지를 사용자에게 표시하는 것이 좋다. 예를 들어, 발생한 예외의 유형과 원인에 대한 간단한 설명을 사용자에게 보여줄 수 있다.

```
except Exception as e:
        # 번역 오류가 발생했을 때 "번역 에러" 팝업 표시
        messagebox.showerror("Translation Error", "An error occurred during translation: " + str(e))
```

4 버튼 및 입력 상태 관리: 번역 중에는 다른 버튼을 비활성화하고 번역이 완료되면 다시 활성화하는 방식으로 사용자 경험을 개선할 수 있다.

```
# 번역 중에 다른 버튼 비활성화
            self.translate_button.config(state="disable")
            self.copy_button.config(state="disable")
            self.lang_combobox.config(state="disable")
# 번역이 끝나면 다른 버튼 다시 활성화
            self.translate_button.config(state="normal")
            self.copy_button.config(state="normal")
            self.lang_combobox.config(state="readonly")
```

5. Test

예문들을 통해 성능에 대한 테스트를 진행했다. 또한 UI가 정상적으로 잘 작동 되는지도 테스트를 진행했다.

예문 1: Chow et. al의 "White-Box Cryptography and an AES Implementation"라는 논문의 Abstract (공백 포함: 1010자, 공백 제외: 872자)

예문 2: 소설 '위대한 개츠비'의 첫 문단

먼저 예문 2의 번역결과에 대한 초벌 번역본과 교정본에 대한 비교를 해보았다.

■ *초벌 번역본: 젊고 더 취약한 몇 년 동안 아버지는 나에게 내가 그 이후로 내 마음 속에서 뒤집어 놓은 조언을 주었다.'T는 당신이 가진 장점이있었습니다. "*

■ *교정본: 젊고 취약한 몇 년 동안, 아버지는 나에게 내 마음 속에서 뒤집어 놓은 조언을 주었습니다. "당신이 가진 장점을 기억하라"는 것이었습니다.*

초벌 번역본은 사람들이 일반적으로 사용하지 않는 부자연스러운 문장 구성을 가 지고 있고 등장인물의 대사를 제대로 번역하지 못했다. 교정본은 그나마 문 장 구성이 자연스럽다. 또한 등장인물의 대사는 중간이 사라지고 다음 대사만 잘 번역이 되었다.

다음은 사전지식이 필요한 긴 글에 대한 번역을 진행해보았다.

■ 초벌 번역본: 암호화 알고리즘의 기존 소프트웨어 구현은 적대적인 사용자가 실행 환경을 제어할 수 있거나 악의적 인 소프트웨어와 공동으로 배치된 곳에서 완전히 불안하지 않습니다. 그러나 현재 추세는 환경에서 사용량이 증가하는 것을 지적합니다. 우리는 신뢰할 수 없는 실행 환경에서 화이트 박스 (Total Access) 공격에 대한 실질적인 보호 수준을 제공하기위 한 암호화된 작용 방법에 대해 논의합니다. 예를 들어, AES가 키 의존 테이블에서 일련의 조회로 구현될 수 있는 방법을 보여줍니다. 의도는 테이블을 개별 단계보다는 조성물을 나타내는 임의의 바이올리로 인코딩하고 포함 된 애플리케이션으로 더 밀어서 암호화 경 계를 확장하여 키를 숨기는 것입니다. 우리는 AES 구현을 부분적으로 정당화하고 권장 구 현의 일부를 제거하면 AES 서브 베이즈 테이블의 패턴을 사용하는 것을 포함하여 지정된 공격을 허용하는 방법을 보여줌으로써 설계에 동기를 부여합니다.

■ 교정본: 기존 암호화 알고리즘 소프트웨어 구현은 적대적인 사용자가 실행 환경을 제어하거나 악의적인 소프트웨어가 공동으로 배치된 곳에서 완벽하게 보호되지 않을 수 있습니다. 하지만 현재 추세는 사용량이 증가하는 환경에서 화이트박스 공격에 대한 실질적인 보호 수준을 제공하기 위해 암호화된 조치방법에 대해 논의하고 있습니다. 예를 들어, AES는 키 의존 테이블에서 일련의 조회로 구현될 수 있는 방법을 보여주고, 이를 통해 테이블을 개별 단계보다는 조각들로 인코딩하여 애플리케이션에 포함시켜 암호화 경계를 확장하고 키를 숨기는 것이 목적입니다. 우리는 AES 구현을 일부 정당화하고 권장 구현의 일 부를 제거하면 AES 서브 베이즈 테이블의 패턴을 사용하여 지정된 공격을 허용하는 방 법을 보여주고, 이를 통해 설계에 동기를 부여합니다.

초벌 번역본 또한 문장 구성이 사람들이 기존에 쓰던 것과 다르게 부자연스럽다. 그런데 단어나 동사 또한 사전지식 없이 그대로 번역만을 한 느낌이 강하다. 그렇기 때문에 인공지능의 검색과 학습 능력을 활용하여 번역기에 사전 지식을 넣고자 했고 chatgpt API를 이용하여 교정을 해보았다. 교정본을 보면 확실히 문장이 읽는데 불편함이 없고 중간중간 어색했던 단어들을 눈치껏 잘 교정한 모습을 볼 수 있었다.

6. Evaluation

Googletrans는 비공식 API였다. 구글에서 정식으로 운영하는 API(Cloud Translation)는 유료이기 때문에 무료인 Googletrans를 사용했었다. 여기다가 Openai의 API인 Chatgpt를 사용하여 Googletrans의 번역 결과 성능을 향상시킬 수 있었다.

번역기 UI도 예외처리를 통해서 어느정도 완성도가 이전보다 높게 나왔고 다만 아쉬운 점은 UI 제작을 처음 해보았기 때문에 꾸미기는 하지 못해 아쉽다. 그래도 1인 프로젝트로 계획했던 것을 잘 완료한 것 같아 뿌듯하다.

Reference

[1] https://pypi.org/project/googletrans/

[2] https://platform.openai.com/docs/api-reference

[3] https://link.springer.com/content/pdf/10.1007/3-540-36492-7_17.pdf

[4] https://m.blog.naver.com/animus98/221658335880

[5] https://pearlluck.tistory.com/372