



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2024-0029461
(43) 공개일자 2024년03월05일

(51) 국제특허분류(Int. Cl.)
G06F 40/58 (2020.01) G06F 40/205 (2020.01)
G06F 40/242 (2020.01) G06N 3/08 (2023.01)
(52) CPC특허분류
G06F 40/58 (2020.01)
G06F 40/205 (2020.01)
(21) 출원번호 10-2022-0107956
(22) 출원일자 2022년08월26일
심사청구일자 2022년08월26일

(71) 출원인
동서대학교 산학협력단
부산광역시 사상구 주례로 47(주례동,
동서대학교)
(72) 발명자
황기현
부산광역시 해운대구 반여로 67 해운대메가센텀한
화꿈에그린아파트 109동 2203호
주수민
부산광역시 사상구 하신중앙로 180 신익강변타운
아파트 106동 1301호
김현주
부산광역시 사상구 주례로 38
(74) 대리인
김석계

전체 청구항 수 : 총 3 항

(54) 발명의 명칭 사투리 자동 번역 시스템

(57) 요약

본발명은 사투리 자동 번역 시스템에 관한 것으로, 번역 모델 개발을 위한 학습데이터 수집모듈;

사전과 번역모델 모듈;

문장인식을 위한 자연어 처리모듈;을 포함하여 구성되는 것을 특징으로 하는

학습데이터 수집모듈은 각 지방에서 사용되는 말뭉치를 수집하고, 인공지능 학습에 사용할 수 있는 json 파일의 데이터셋을 구축하는 것으로,

본발명은 사투리를 빠르고 정확하게 번역하여 제공하는 현저한 효과가 있다.

대표도 - 도1



(52) CPC특허분류

G06F 40/242 (2020.01)

G06N 3/08 (2023.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	1345356165
과제번호	LINC3.0-2022-60
부처명	교육부
과제관리(전문)기관명	한국연구재단
연구사업명	3단계 산학협력 선도대학 육성사업(LINC 3.0)
연구과제명	3단계 산학협력 선도대학 육성사업(LINC 3.0)(수요맞춤성장형)
기 여 율	1/1
과제수행기관명	동서대학교산학협력단
연구기간	2022.03.01 ~ 2028.02.28

명세서

청구범위

청구항 1

번역 모델 개발을 위한 학습데이터 수집모듈;

사전과 번역모델 모듈;

문장인식을 위한 자연어 처리모듈;을 포함하여 구성되는 것을 특징으로 하는

학습데이터 수집모듈은

각 지방에서 사용되는 말뭉치를 수집하고, 인공지능 학습에 사용할 수 있는 데이터셋을 구축하는 것을 특징으로 하는 사투리 자동 번역 시스템

청구항 2

제1항에 있어서, 상기 사전과 번역모델 개발모듈은 번역을 위해 단어의 정보를 담은 사전을 만들고, 구분을 위해 사투리 사전을 만들며, 전체 사전을 API로 불러온 후, 단어에 따라 번역 모델에 넣거나 그대로 사용하는 것을 특징으로 하는 사투리 자동 번역 시스템

청구항 3

제2항에 있어서, 상기 문장인식을 위한 자연어 처리모듈은 문장을 어절로 나누고, 불용어나 기호등을 제거하여 어절 단위로 분리하는 것을 특징으로 하는 사투리 자동 번역 시스템

발명의 설명

기술 분야

[0001] 본발명은 사투리 자동 번역 시스템에 관한 것으로, 보다 상세하게는 사투리를 빠르고 정확하게 번역하여 제공하는 사투리 자동 번역 시스템에 관한 것이다.

배경 기술

[0002] 일반적으로 사투리를 표준어처럼 알아듣게 번역하는 시스템이 개발되고 있으며, 종래특허기술의 일례로서 등록특허공보 등록번호 10-1836430호에는 단말 장치에서 사투리가 섞인 음성을 인식하여 번역하기 위한 방법으로서,

[0003] (a) 발생된 음성을 입력 인터페이스를 통해 인식하는 단계;

[0004] (b) 상기 인식된 음성이 사투리가 섞인 발생 음성에 해당되면, 상기 인식된 사투리 음성이 갖고 있는 음색 노이즈를 제거하는 단계;

[0005] (c) 상기 음색 노이즈가 제거된 사투리 언어 데이터를 자국의 표준 언어 데이터로 변환하는 단계;

[0006] (d) 상기 변환된 표준 언어 데이터를 번역기를 통해 소정의 타국어로 번역하는 단계; 및

[0007] (e) 상기 번역된 타국어를 출력 인터페이스를 통해 표시 화면에 표시하거나 음성 형태로 출력하는 단계;

[0008] 를 포함하는 음성을 인식하여 번역하는 방법에 있어서,

[0009] 상기 (b) 단계는,

[0010] 상기 인식된 음성에 대한 표준어의 제2 음색 수치와 상기 사투리 음성의 제1 음색 수치를 비교한 언어 식별 정확도를 기초로, 상기 표준어의 음색보다 큰 상기 사투리 음성의 음색 노이즈를 제거하는 단계; 및

[0011] 상기 음색 노이즈가 제거된 음성에 대해 음소 노이즈를 더 제거하는 단계;

[0012] 를 더 포함하며, 상기 제1 음색 수치는, 상기 사투리가 갖는 정해진 음색의 높낮이 크기이며, 상기 제2 음색 수치는, 상기 사투리에 매칭된 표준어가 갖는 정해진 음색의 높낮이 크기이고, 상기 언어 식별 정확도는, 상기 인

식된 음성이 포함하는 각 단어 또는 문장마다 상기 제1 음색 수치와 제2 음색 수치를 비교하여 결정되며, 상기 음소 노이즈는 쌍자음 발음인 ㅅ 및 ㅂ에 대한 노이즈인, 음성을 인식하여 번역하는 방법이 공개되어 있다.

[0013] 또한, 등록특허공보 등록번호 10-2406251호에는 언어 현지화에서 번역 완성도를 높이기 위한 자동 번역 및 코딩 방법이 공개되어 있다.

[0014] 그러나 상기 종래기술들은 번역이 정확하지 않고 번역속도가 느린 단점이 있었다.

발명의 내용

해결하려는 과제

[0015] 따라서 본발명은 상기와 같은 문제점을 해결하기 위하여 안출된 것으로, 한국어 번역 특성을 고려하여 ETRI 인공지능 API를 사용하고 DNN, ReLU기반의 번역 모델을 사용하는 번역 시스템을 제공하고자 하는 것이다.

과제의 해결 수단

[0016] 본발명은 사투리 자동 번역 시스템에 관한 것으로, 번역 모델 개발을 위한 학습데이터 수집모듈;

[0017] 사전과 번역모델 모듈;

[0018] 문장인식을 위한 자연어 처리모듈;을 포함하여 구성되는 것을 특징으로 하는

[0019] 학습데이터 수집모듈은 각 지방에서 사용되는 말뭉치를 수집하고, 인공지능 학습에 사용할 수 있는 json 파일의 데이터셋을 구축하는 것을 특징으로 한다.

발명의 효과

[0020] 따라서 본발명은 사투리를 빠르고 정확하게 번역하여 제공하는 현저한 효과가 있다.

도면의 간단한 설명

[0021] 도 1은 본발명의 번역 모델 구성도

도 2는 데이터셋 구축 과정도

도 3은 사전 활용에 이용할 데이터 구조도

도 4는 과정 순서도

도 5는 의존구문분석 ETRI API 사용 예시도

도 6은 예상 결과물 도면

도 7은 JSON파일 중 일부 화면

도 8은 utterance 중 방언 어절 리스트화면

도 9는 [라벨]경상도_학습데이터_1 파일 구성, txt파일, json파일화면

도 10은 pickle파일 생성과 생성된 모습화면

도 11은 인텍스를 이용한 방언-표준어 쌍 화면

도 12는 pairs.csv 파일화면

13은 번역생성화면

14는 번역결과화면

도 15는 품사구분을 위한 태그 선언화면

도 16은 train_df.csv : 단어의 품사를 저장한 파일 화면

도 17은 동의어 학습을 위해 추가된 리스트 화면

도 18은 pickle 파일 (eojeol_dict.pickle, synonym_list_test.pickle)

도 19는 동의어 모델을 학습시키는 화면(synonym_model.h5)

도 20은 summary함수를 사용하여 모델의 구조 및 노드의 개수 등 정보를 표시해주는 화면

도 21은 ETRI API KEY 발급화면

도 22는 API를 사용하기 위한 파라미터 작성화면

도 23은 번역이 제대로 되는 경우화면

도 24는 데이터셋 내 단어가 존재하지 않는 모습과, 존재할 때 단어 번역 결과화면

발명을 실시하기 위한 구체적인 내용

- [0022] 본발명은 사투리 자동 번역 시스템에 관한 것으로, 번역 모델 개발을 위한 학습데이터 수집모듈;
- [0023] 사전과 번역모델 모듈;
- [0024] 문장인식을 위한 자연어 처리모듈;을 포함하여 구성되는 것을 특징으로 하는
- [0025] 학습데이터 수집모듈은
- [0026] 각 지방에서 사용되는 말뭉치를 수집하고, 인공지능 학습에 사용할 수 있는 json 파일의 데이터셋을 구축하는 것을 특징으로 한다.
- [0027] 또한, 상기 사전과 번역모델 개발모듈은 번역을 위해 단어의 정보를 담은 사전을 만든다.
- [0028] 구분을 위해 사투리 사전을 만들고, 전체 사전을 API로 불러온 후, 단어에 따라 번역 모델에 넣거나 그대로 사용하는 것을 특징으로 한다.
- [0029] 또한, 상기 문장인식을 위한 자연어 처리모듈은
- [0030] 문장을 어절로 나누고, 불용어나 기호등을 제거하여 어절 단위로 분리하는 것을 특징으로 한다.
- [0032] 본발명을 첨부도면에 의해 상세히 설명하면 다음과 같다.
- [0033] 번역 모델 개발을 위한 학습데이터 수집단계는 제어부는 학습데이터 수집모듈을 통해 각 지방에서 사용되는 말뭉치를 수집하고, 인공지능 학습에 사용할 수 있는 json 파일의 데이터셋을 구축한다.
- [0034] 용도에 맞게 활용하기 위하여 데이터에서 필요한 정보를 리스트화 한다.
- [0036] 사전과 번역모델 구축단계는 제어부는 사전과 번역모델 구축 모듈을 통해 번역을 위해 단어의 정보를 담은 사전을 만든다.
- [0037] 구분을 위해 사투리 사전을 만들고, 전체 사전을 API로 불러온 후, 단어에 따라 번역 모델에 넣거나 그대로 사용한다.
- [0038] 사투리 번역 모델 : 입력 문장 안에서 사투리 단어를 발견하면 번역모델로 보낼 수 있도록 한다.
- [0039] 동음이의어 분류 모델 : 단어를 어절단위로 구분하기 때문에 같은 텍스트여도 다른 뜻일 수 있음. 입력받은 동의어가 사투리인지 표준어인지 구분하는 모델을 생성한다.
- [0040] 문장인식을 위한 자연어 처리단계는 제어부는 자연어 처리모듈을 통해 문장을 어절로 나누고, 불용어나 기호등을 제거하여 어절 단위로 분리한다.
- [0041] 동음이의어일 경우 품사를 구분하여 같은 단어의 뜻을 구분한다.
- [0043] 상기 번역 모델 구축을 위한 학습데이터 수집단계는 구체적으로, 사투리를 사용하는 10대 ~ 60대의 일상 대화를 수집하여 음성을 문자로 실시간 변환한다. 사투리 AI 학습 데이터 구축을 위하여 수집한 사투리를 일차적으로 지역적 사투리의 특성을 살려 사투리를 전사하고 표준에 규정에서 벗어나 방언에 해당하는 부분에 대한 표준어

대응쌍을 제시하는 형태로 전사하여 만들어진 JSON 형식의 학습 데이터셋을 사용한다.

[0044] 학습데이터 JSON 파일에서 말뭉치에 해당하는 utterance 부분을 리스트로 만들어 사투리 사전을 위한 데이터로 사용한다(방언 어절 eojeol, 표준어 어절 standard).

[0045] 도별 '한국어 방언 발화' 데이터셋을 번역모델 학습을 위한 데이터로 사용하여 사투리 번역기로 통합한다.

[0047] 사전과 번역모델 구축은 구체적으로 사투리, 표준어 사전 생성 함수로서, 사전 생성을 위해 학습데이터 중 utterance 태그가 되어 있는 대화 내용을 불러온다. re 내장모듈 내 compile 메소드를 사용해 학습데이터 중 문장만을 가져온다.

[0048] 문장 중 각 단어의 태그를 통해 사투리는 dialect 리스트로, 표준어는 standard 리스트로 저장한다. (eojeol, dialect, standard)

[0049] 높은 용량의 텍스트 파일을 파싱할 경우 txt파일은 비효율적이기 때문에 필요한 딕셔너리, 리스트, 튜플을 pickle을 통해 저장한다. (pickle : 필요한 딕셔너리, 리스트, 튜플을 pickle을 통해 저장한다. pickle 모듈을 활용해 객체 자체를 바이너리로 저장함)

[0050] - 방언 -> 인덱스, 표준어 -> 인덱스, 인덱스->방언, 표준어->방언을 구분하여 pickle.dump를 이용하여 사용한다.

[0052] 동의어 처리 데이터셋은 같은 단어도 문장에서 쓰임이 다르면 뜻이 달라짐을 이용해, 품사가 다름을 통해 동음이의어를 구분한다.

[0053] 품사는 의존 구문분석 태그셋의 구문레이블 ['NP', 'VP', 'AP', 'VNP', 'DP', 'IP', 'X', 'L', 'R']), 기능레이블(['SBJ', 'OBJ', 'MOD', 'AJT', 'CMP', 'CNJ']) 태그를 이용해 정의하고, 단어에 품사 태그를 적용하여 동음이의어를 구분하는데 사용할 수 있게 만든다. (의존관계 설정 가이드라인은 세종 구문분석 가이드라인을 기반으로 한 TTA 표준 의존 구문분석 가이드라인 (TTAK.K0-10.0853)과 제27회 한글 및 한국어 정보처리 학술대회 논문집의 "의존 구문분석을 위한 한국어 의존관계 가이드라인 및 엑소브레인 언어분석 말뭉치"를 따르고 있다.)

[0055] 동의어 사전 생성 함수는 방언 사전과 표준어 사전에 모두 있는 단어를 동의어 리스트에 저장한다. 리스트에 저장한 내용을 더 빠르게 사용하기 위해 pickle.dump를 사용하여 저장한다.

[0056] 단어가 사투리, 표준어, 동음이의어인지 구분하기 위해 단어와 연결된 단어의 품사를 ETRI API를 이용해 가져온다.

표 1

		사투리 사전	
		0	X
표준어 사전	0	동의어	표준어
	X	사투리	불용어

[0058] <표> 사투리와 표준어 구분

[0059] 동의어 처리 모델은 문장을 이어붙이고, 이어붙인 문장의 어절을 돌면서 동의어 목록에 있는 어절을 발견하면 동의어를 학습시킬 리스트에 추가한다.

[0060] 동의어 모델을 학습시킨다. (학습 모델 이름 : synonym_model.h5)

[0061] relu 함수 : 내부 hidden layer를 활성화 시키는 함수로 sigmoid를 사용하지 않고 ReLU라는 활성화 함수를 사용하게 되는데, 이 함수는 쉽게 말해 0보다 작은 값이 나온 경우 0을 반환하고, 0보다 큰 값이 나온 경우, 그 값을 그대로 반환하는 함수다. 0보다 큰 값일 경우 1을 반환하는 sigmoid와 다르다. 따라서 내부 hidden layer에는 ReLU를 적용하고, 마지막 output layer에서만 sigmoid 함수를 적용하면 이전에 비해 정확도가 훨씬 올라가게 된다.

- [0062] 문장을 입력하면 어절별로 나누어서 정리한 후, 만약 사투리 사전에 있으면 전체 어절 사전으로 넘어가고 전체 어절 사전에도 존재하면 동의어 처리 모델로 넘겨주어서 표준어에 가까운지 사투리에 가까운지 판단한 후 출력한다.
- [0064] 문장을 어절로 나누거나, 품사 정보를 가져오기 위해 ETRI open API 이용한다.
- [0065] 결과물로서, 번역하길 원하는 문장을 입력하고 각 어절별로 문장을 인식하여 문장의 품사, 전체 어절, 동의어, 어절별 의존 구문의 품사에 따라 사전을 구축한다. 가장 정확성이 높은 문장을 계산하여 출력한다.
- [0067] 본발명에 대한 실험결과는 다음과 같다.
- [0068] 번역 모델 개발을 위한 학습데이터 수집단계는 학습데이터 수집 및 정제
- [0069] 사투리와 표준어 대응쌍으로 전사된 JSON 파일에서 말뭉치에 해당하는 utterance 부분을 리스트로 만들어 사투리 사전을 위한 데이터로 사용함 (방언 어절 eojeol, 표준어 어절 standard)
- [0070] 방언과 표준어 전사 쌍을 태그로 분류하여 리스트로 저장한다. (eojeol, dialect, standard)
- [0072] 사전과 번역모델 개발단계;
- [0073] 사투리, 표준어 사전 생성 함수
- [0074] 솔트룩수에서 주관하여 구축한 학습데이터셋을 이용한다. 사전 생성을 위해 학습데이터 중 utterance 태그가 되어 있는 대화 내용을 불러온다.
- [0075] `re.compile('[^가-힣]+')` 을 이용해 utterance 중 문장을 가져온다.
- [0077] 문장 중 각 단어의 태그를 통해 사투리는 dialect 리스트로, 표준어는 standard 리스트로 저장한다. (eojeol, dialect, standard)
- [0078] 방언 -> 인덱스, 표준어 -> 인덱스, 인덱스->방언, 표준어->방언을 구분하여 pickle.dump를 이용하여 사용한다.
- [0079] 인덱스를 이용해 방언과 표준어 쌍을 만든다. 방언 표준어 쌍을 csv 파일로 저장한다.
- [0080] 사투리 to 표준어 번역 모델 개발은 방언 표준어 쌍(Paris.csv)을 이용해 사투리의 표준어 번역을 학습시킨다. 자주 사용된 사투리일수록 정확도가 높다
- [0081] relu함수를 이용하여 패키지를 부착한 후 역전파를 차용하여 가장 확률이 근접한 값을 출력한다.
- [0082] Sequential()함수를 사용하여 모델을 생성한다.
- [0083] 번역모델을 학습한 후 translate_model.save() 함수를 통해 번역모델을 저장한다.
- [0085] 동의어 처리 데이터셋은
- [0086] 품사는 의존 구문분석 태그셋의 구문레이블 (a_tag), 기능레이블([b_tag]) 태그를 이용해 정의하고, 단어에 품사 태그를 적용하여 동음이의어를 구분하는데 사용할 수 있게 만든다.
- [0087] a, b, c 태그를 생성하여서 각각 어절별로 나누어서 할당해 준다.
- [0088] 문장을 이어붙이고, 이어붙인 문장의 어절을 돌면서 동의어 목록에 있는 어절을 발견하면 동의어를 학습시킬 리스트에 추가한다.
- [0089] train_df.csv : 단어의 품사를 저장한 파일.
- [0090]

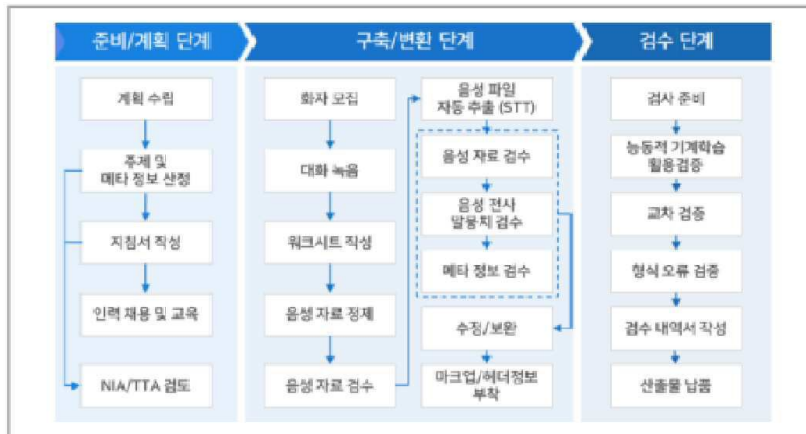
- [0091] 동의어 사전 생성 함수는 품사 태그를 선언하고 단어의 품사가 다를 수 있도록 동음이의어를 구분한다. 단어가 사투리인지 표준어인지 구분하기 위해 단어와 연결된 단어의 품사를 ETRI API를 이용해 가져온다.
- [0092] 사투리->인덱스, 인덱스->표준어 pickle 파일을 이용해 데이터 전체 어절을 딕셔너리로 만든다. 문장 내에서 표준어와 사투리를 구분하기 위해 사투리 인덱스에는 0을 붙여준다. (eojeol_dict.pickle)
- [0093] - 방언 사전과 표준어 사전에 모두 있는 단어를 동의어 리스트에 저장한다. 리스트에 저장한 내용을 더 빠르게 사용하기 위해 pickle.dump를 사용하여 저장한다. (synonym_list_test.pickle)
- [0095] 동의어 처리 모델 개발은 동의어 모델을 학습시킨다. (synonym_model.h5)
- [0096] summary함수를 사용하여 모델의 구조 및 노드의 개수 등 정보를 표시해준다.
- [0098] 문장인식을 위한 자연어 처리 (ETRI)는 입력하면 어절별로 나누어서 정리한 후 사투리 사전에 있으면 전체 어절 사전으로 넘어가고, 전체 어절 사전에도 존재하면 동의어 처리 모델로 넘겨주어서 표준어에 가까운지 사투리에 가까운지 판단한 후 출력한다.
- [0100] 결과물로서 목표 문장을 입력하고 번역 버튼을 누르면 번역결과를 볼 수 있다.
- [0101] 번역이 제대로 되는 경우는
- [0102] 사투리 문장 → 표준어 문장 [아이가 → 앓아]
- [0103] 표준어 문장 → 표준어 문장 [아이가 → 아이가]
- [0105] 번역이 잘되지 않은 경우는 데이터셋 내에 존재하지 않거나 사용된 빈도수가 저조한 경우 제대로 된 단어의 의미 구분을 할 수 없어 번역이 원활하게 진행되지 않는다.
- [0106] 본발명 제어기는 발성한 음성을 입력받는 입력 인터페이스; 사투리 언어 데이터를 포함한 언어 데이터를 저장하는 저장부; 이하의 제어기의 명령에 대응하여 해당하는 데이터를 출력하는 출력 인터페이스; 및 상기 입력 인터페이스와 상기 저장부 및 상기 출력 인터페이스를 제어하는 제어기를 포함하여 구성된다.

도면

도면1



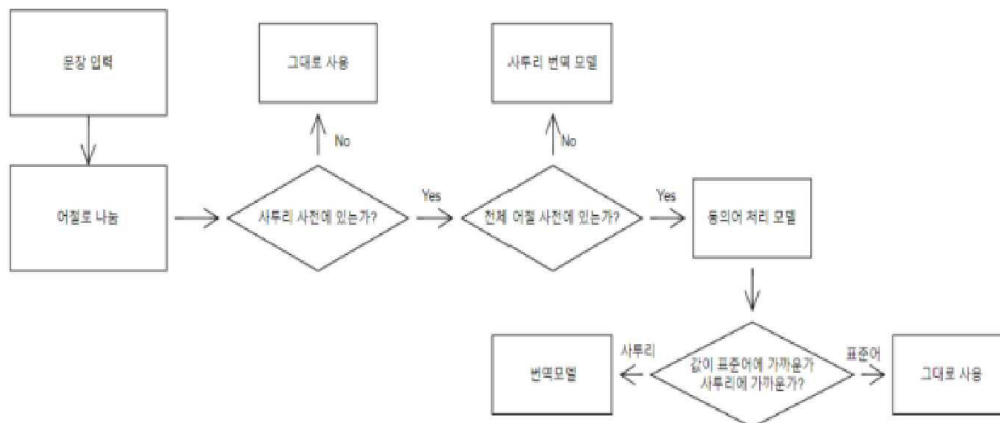
도면2



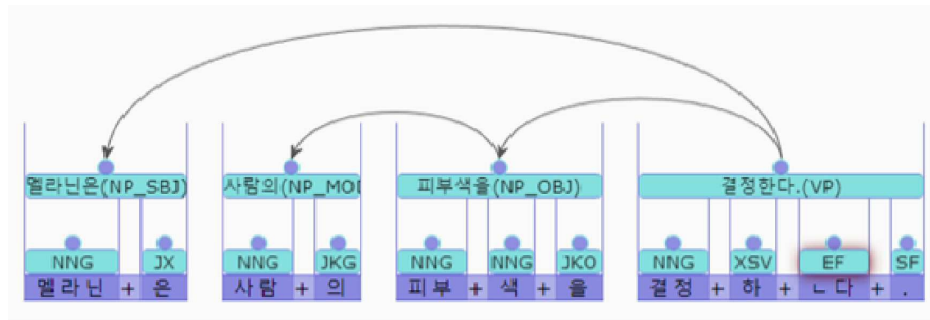
도면3

utterance	eojeolList		array(object)		방언 어절 단위 정보
		id	num	Y	
		eojeol	string		방언 어절
		standard	string		표준어 어절
		isDialect	boolean		방언 어절 여부

도면4

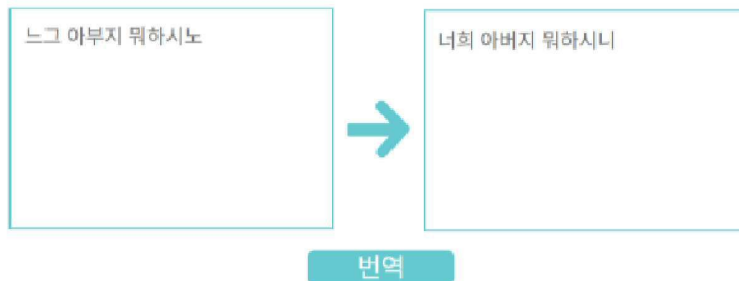


도면5



도면6

사투리 번역기



도면7

```

    },
    "utterance": [
      {
        "id": "DKC120000001.1.1.1",
        "form": "자 음식을 멀 좋아하느냐 하면은",
        "standard_form": "자 음식을 멀 좋아하느냐 하면은",
        "dialect_form": "자 음식을 멀 좋아하느냐 하면은",
        "speaker_id": "1",
        "start": 0.01,
        "end": 3.45,
        "note": "",
        "eojeollList": [
          {
            "id": 1,
            "eojeol": "자",
            "standard": "자",
            "isDialect": false
          },
          {
            "id": 2,
            "eojeol": "음식을",
            "standard": "음식을",
            "isDialect": false
          }
        ]
      }
    ]
  }

```

도면8

['겁나', '금까', '가까교', '니', '싫다매', '아이다', '그니까', '먹
었나', '뵈었니까', '사바리', '받을라고', '또마냥게', '멕일라고',
'고맙디', '얹았나', '왔나', '없나', '글로', '내', '지밖에', '지',
'내한테', '사나', '기지배가', '쌩판', '한다', '아이가', '아나', '니
는', '시제', '오빠야랑', '그까', '니가', '그면', '강', '임마', '니
라고', '지가', '간다', '꺼가', '지는', '울그락불그락', '니도', '했
다', '시원하드나', '바까야', '되나', '싫다', '하겠나', '갈나', '온
디', '온나', '가시나', '느그', '산다', '쫘', '내랑', '인연인갑다',
'고거는', '기집애도', '못해나', '또금', '속하게', '기냥', '조끔',
'된디', '에나콩떡', '안됐는디', '또금', '그런갑다', '그리고', '아닌
갑다', '따른', '이케', '그까지', '그라고', '요즘', '놀랬던', '요령
게', '묵으러', '땡기는', '그기', '짜지', '뺏기고', '묵을', '인자',
'내리놓고', '썩어서', '인저', '그래', '하이튼', '땡에', '짜잘짜잘',
'걸거치니까', '디게', '맞차가지고', '꽤안았다', '주소', '사겨',
'저', '짜은', '짜고', '이카면서도', '깨운하고', '지끔은', '어데까
지', '그래가', '먹고예', '그러까네', '그카고', '이카면서', '그러께
네', '카고', '아파가', '이자', '맞더라고예', '카는데', '가가', '와
가', '우웠든', '티나는', '해야지예', '카는', '꼬꾸장하게', '아이
고', '마이', '있잖아예', '뽕고', '뽕어가', '있더라고예', '돼가',
'꾸웠다든지', '꾸워서', '쌩', '쌩으로', '쌩두부를', '쌩두부', '섞어
서', '하더라고예', '백에', '뽕어', '카면', '그카다', '지금또', '쌩
서', '카다가', '많았어예', '저그', '아는', '엮드려가', '거라요',

도면9

DKCI20000001.json
DKCI20000001.txt
DKCI20000002.json
DKCI20000002.txt
...
DKSR20007343.json
DKSR20007343.txt

아직은 (인제)/(이제) 뭐~ 상대방분님께서
아주 어린 아이고 어린 나이 때는 뭐~ 말을
잘 들을 수도 있고 안 들을 수도 있고
(고걸)/(그걸) 이제 어떻게 다루느냐 문젠데
제가 지금 하고 있는 일이 뭐냐 하면은
(인제)/(이제) 가게를 그만 두고 한 삼 년 쉬
었다가 어~ 오후에 세 시간만 네 시부터 일
곱 시까지

```
{
  "id": "DKSR20007329.1.1.110",
  "form": "아직은 (인제)/(이제) 뭐~ 상대방분님께",
  "standard_form": "아직은 이제 뭐~ 상대방분님께",
  "dialect_form": "아직은 인제 뭐~ 상대방분님께",
  "speaker_id": "2",
  "start": 781.88,
  "end": 791.07,
  "note": "",
  "eojeolList": [
    {
      "id": 1,
      "eojeol": "아직은",
      "standard": "아직은",
      "isDialect": false
    },
    {
      "id": 2,
      "eojeol": "인제",
      "standard": "이제",
      "isDialect": true
    }
  ]
}
```

도면10

```

with open('dialect_to_index.pickle', 'wb') as fw:
    pickle.dump(dialect_to_index, fw)

with open('standard_to_index.pickle', 'wb') as fw:
    pickle.dump(standard_to_index, fw)

with open('index_to_dialect.pickle', 'wb') as fw:
    pickle.dump(index_to_dialect, fw)

with open('index_to_standard.pickle', 'wb') as fw:
    pickle.dump(index_to_standard, fw)

with open('dialect_to_index.pickle', 'rb') as fr:
    dialect_to_index = pickle.load(fr)

with open('standard_to_index.pickle', 'rb') as fr:
    standard_to_index = pickle.load(fr)

with open('index_to_dialect.pickle', 'rb') as fr:
    index_to_dialect = pickle.load(fr)

with open('index_to_standard.pickle', 'rb') as fr:
    index_to_standard = pickle.load(fr)

```

이름

- ☐ standard_to_index.pickle
- ☐ dialect_to_index.pickle
- ☐ index_to_dialect.pickle
- ☒ index_to_standard.pickle

도면11

```

In [4]: pairs = make_pair(raw_data)
...: pairs[:10]
Out[4]:
  dialect_words  standard_words
0             0                0
1             1                1
2             2                2
3             3                3
4             4                4
5             1                1
6             5                5
7             6                1
8             1                1
9             7                7

In [5]:

```

도면12

	A	B	C	D	E
1		dialect_wo	standard_words		
2	0	0	0		
3	1	1	1		
4	2	2	2		
5	3	3	3		
6	4	4	4		
7	5	1	1		
8	6	5	5		
9	7	6	1		
10	8	1	1		
11	9	7	7		
12	10	6	1		
13	11	8	8		
14	12	9	9		
15	13	0	590		
16	14	0	0		

도면13

```
trans_xtrain = np_utils.to_categorical(pairs["dialect_words"])
trans_ytrain = np_utils.to_categorical(pairs["standard_words"])
len(trans_xtrain[0])

translate_model = Sequential()
translate_model.add(Dense(256, input_dim = 7581, activation="relu"))
translate_model.add(Dense(128, input_dim = 7581, activation="relu"))
translate_model.add(Dense(7581, activation="softmax"))
translate_model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
translate_model.fit(trans_xtrain, trans_ytrain, batch_size=100, epochs=50)
```

도면14

```
In [11]: translate_model.fit(trans_xtrain, trans_ytrain, batch_size=100, epochs=50)
Epoch 1/50
85/1193 [=>.....] - ETA: 55s - loss: 5.7962 - accuracy: 0.4707
```

```
accuracy: 0.9793
Epoch 50/50
1193/1193 [=====] - 59s 49ms/step - loss: 0.0525 - accuracy: 0.9794
```

```
Out[11]: <keras.callbacks.History at 0x10edb181fd0>
```

```
In [12]: translate_model.save("translate_model_1.h5")
```

```
In [31]: translate("이제")
```

```
0
```

```
Out[31]: '이제'
```

도면15

```
# 품사 사전
a_tag = ['NP', 'VP', 'AP', 'VNP', 'DP', 'IP', 'X', 'L', 'R']
b_tag = ['SBJ', 'OBJ', 'MOD', 'AJT', 'CMP', 'CJ']
c_tag = []
pos_dict = {}
```

도면16

```
train_df.to_csv("train_df_1_word_index_col_test.csv")
train_df = pd.read_csv("train_df_1_word_index_col_test.csv")
train_df = train_df[["pos", "word", "word_index_col"]]
synonym_xtrain = np_utils.to_categorical(train_df["pos"])
synonym_ytrain = np_utils.to_categorical(train_df["word_index_col"])
```

```
train_df.to_csv("train_df_1.csv")
train_df = pd.read_csv("train_df.csv")
train_df = train_df[["pos", "word"]]
len(set(train_df["word"]))
```

도면17

	A	B	C	D		A	B	C	D	E
1		pos	word		1		pos	word	word_index_col	
2	0	0	1594		2	0	0	1594	0	
3	1	3	1594		3	1	3	1594	0	
4	2	0	9192		4	2	0	9192	1	
5	3	54	8394		5	3	54	8394	2	
6	4	59	218		6	4	59	218	3	
7	5	8	8652		7	5	8	8652	4	
8	6	56	9192		8	6	56	9192	1	
9	7	55	11028		9	7	55	11028	5	
10	8	3	9192		10	8	3	9192	1	
11	9	55	6068		11	9	55	6068	6	
12	10	3	9192		12	10	3	9192	1	
13	11	55	10146		13	11	55	10146	7	

도면18

- eojeol_dict.pickle
- synonym_list_test.pickle

도면19

```
synonym_model = Sequential()
synonym_model.add(Dense(512, input_dim = 61, activation="relu"))
synonym_model.add(Dense(256, activation="relu"))
synonym_model.add(Dense(128, activation="relu"))
synonym_model.add(Dense(3961, activation="softmax"))
synonym_model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
synonym_model.fit(synonym_xtrain, synonym_ytrain, batch_size=500, epochs=50)

synonym_model.save("synonym_model.h5")
```

도면20

```
In [42]: synonym_model.summary()
Model: "sequential_1"
```

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 512)	31744
dense_4 (Dense)	(None, 256)	131328
dense_5 (Dense)	(None, 128)	32896
dense_6 (Dense)	(None, 3961)	510969

```

Total params: 706,937
Trainable params: 706,937
Non-trainable params: 0
```

도면21

STEP2. API KEY 확인

신청 이메일	khj5606kh@gmail.com
키	6a539849-5463-4d19-b8a0-290bb5c96e30
상태	승인 완료
발급일	2022-05-17 08:13:36

도면22

```
# 단어와 연관된 단어의 품사 추출
def get_mods(sentence, num):
    #print(sentence, num)
    openApiURL = "http://aipen.etri.re.kr:8000/WiseNLU"

    accessKey = "aba3ad5d-3318-4061-8a5d-799c8521b082"

    #bd6d2bff-4e44-4035-8591-947fda262051
    analysisCode = "dpase"

    text = sentence

    requestJson = {
        "access_key": accessKey,
        "argument": {
            "text": text,
            "analysis_code": analysisCode
        }
    }

    http = urllib3.PoolManager()
```

도면23



도면24

```
In [18]: translate("뉘'하시노")
Traceback (most recent call last):

Input In [18] in <cell line: 1>
      translate("뉘'하시노")

Input In [18] in translate
      index = dialect_to_index[word]
KeyError: '뉘'하시노'
```

```
In [19]: translate("아이가")
26
Out[19]: '않아'

In [20]: translate("가는")
1262
Out[20]: '개는'
```