|  |  |
| --- | --- |
| 교육 제목 | AI 프로젝트 기반 S/W 전문가 양성과정 |
| 교육 일시 | 22..01.14 |
| 교육 장소 | 집 |
| **교육 내용** | |
| 오전 | 1. 2022.1.13 review 2. Tokenization 이어서 3. 공백 기반 토큰화 문장을 어떤 기준으로 쪼갰을 때, 쪼개진 각 단어들을 토큰(Token) 이라고 부릅니다. 그리고 그 쪼개진 기준이 토큰화(Tokenization) 기법에 의해 정해지죠. 이번 스텝에서는 토큰화의 여러 가지 기법에 대해 배워보도록 하겠습니다. tokens = corpus.split() 4. 형태소 기반 토큰화 형태소 - 뜻을 가진 가장 작은 말의 단위 한국어 형태소 분석기는 대표적으로 아래 두 가지가 사용 KoNLPy 파이썬 한국어 패키지, kakao/khaiii 5. Tokenization Summary 한국어의 경우 접사를 분리하여 희소성을 낮추고 띄어쓰기를 통일하기 위해 tokenization을 수행, 굉장히 많은 POS Tagger가 존재하는데, 전형적인 쉬운 문장(표준 문법을 따르며, 구조가 명확한 문장)의 경우, 성능이 비슷함. 하지만 신조어나 고유명사를 처리하는 능력이 다름. 따라서 주어진 문제에 맞는 정책을 가진 tagger를 선택하여 사용해야 함. 6. TOKENIZATION STYLE의 특성 토큰 길이가 짧을수록 – Vocabulary 크기 감소, 희소성 문제 감소 OOV가 줄어듦, Sequence의 길이가 길어짐 - 모델의 부담 증가, 극단적 형태 Char 단위 토큰 길이가 길수록 – Vocabulary 크기 증가 – 희소성 문제 증대, OOV가 늘어남, Sequence의 길이가 짧아짐 – 모델의 부담 감소 🡪 토큰 길이에 따른 Trade off가 존재 7. 정보량에 따른 이상적인 형태 빈도가 높을 경우, 하나의 token으로 나타내고, 빈도가 낮을 경우 더 잘게 쪼개어 각각 빈도가 높은 token으로 구성한다 🡪 압축 알고리즘? 8. 인코딩  단어 기반 인코딩 🡪 단어를 쪼개서 indexing import tensorflow as tf from tensorflow import keras from tensorflow.keras.preprocessing.text import Tokenizer tokenizer = Tokenizer(num\_words = 100) num\_words = 100 🡪 단어 100개 사용 tokenizer.fit\_on\_texts(sentences) # 문자 데이터를 입력받아 리스트의 형태로 변환 word\_index = tokenizer.word\_index # 토큰별 단어에 index를 매핑 9. 텍스트를 시퀀스로 변환하기 indexing한 단어들을 시퀀스로 변환 sequences = tokenizer.texts\_to\_sequences(sentences) # 텍스트를 시퀀스로 변환 10. 패딩 설정하기 from tensorflow.keras.preprocessing.sequence import pad\_sequences padded = pad\_sequences(sequences) |
| 오후 | 1. Tokenizer()의 flags num\_words 🡪 단어 빈도가 많은 순서로 num\_words개의 단어를 보존 filters 🡪 걸러낼 문자 리스트를 적어줌. default : !”@#()^$%^\*@#$ 등 lower 🡪 소문자로 변환. 0 or 1 split 🡪 문자열을 적어줘야하고 단어를 분리하는 기준을 적어줌 char\_level 🡪 1: 모든 문자가 토큰으로 처리, 0은 안함 oov\_token 🡪 값이 지정 되면 text2sequence 호출 과정에서 word\_index에 추가되어 oov vocabulary words로 대체(<OOV>, <UNK>) 2. 공백 기반 토큰화 실습 def tokenize(corpus):  tokenizer = Tokenizer(filters = "") # Tokenizer  tokenizer.fit\_on\_texts(corpus) # 문자 --> 리스트   tensor = tokenizer.texts\_to\_sequences(corpus) # 텍스트 -> 시퀀스  tensor = pad\_sequences(tensor, padding='post') # 패딩처리  return tensor, tokenizer  split\_corpus = [] for kor in filtered\_corpus:  split\_corpus.append(kor.split())  split\_tensor, split\_tokenizer = tokenize(split\_corpus) print("Split Vocab Size :", len(split\_tokenizer.index\_word)) 3. 형태소 기반 토큰화 실습 def mecab\_split(sentence):  return mecab.morphs(sentence)   #mecab 단어장 생성 mecab\_corpus = [] for kor in filtered\_corpus:  mecab\_corpus.append(kor.split())  mecab\_tensor, mecab\_tokenizer = tokenize(mecab\_corpus) print("Mecab Vocab size :", len(mecab\_tokenizer.index\_word)) 4. 한국어를 Tokenization 할 때 공백 기반 Tokenization 절대 지양!!!!! 5. SUBWORD SEGMENTATION 학습 데이터로부터 얻은 모델을 활용하여 똑 같은 분절 수행 6. 단어보다 더 작은 의미 단위 : Subword 많은 언어들에서, 단어는 더 작은 의미 단위들이 모여 구성됨. 따라서 이러한 작은 의미 단위로 분절할 수 있다면 좋은 것 하지만, 이를 위해선 언어별 subword사전이 존재해야 할 것 7. OOV(out-of-vocabulary)또는 UNK(Unknown Token)이라 표현함 8. 서브워드 분리(Subword Segmentation)작업 하나의 단어는 더 작은 단위의 이미있는 여러 서브워드들(birthplace = birth+place)의 조합으로 구성 된 경우가 많기 때문에 하나의 단어를 여러 subword로 분리해서 단어를 인코딩 및 임베딩하겠다는 의도를 가진 전처리 작업 9. 이를 통해 OOV나 희귀단어, 신조어 같은 문제들을 완화 10. OOV가 미치는 영향 입력데이터에 OOV가 발생할 경우, <UNK> 토큰으로 치환하여 모델에 입력 e.g. 나는 학교에 가서 밥을 먹었다. 🡪 나는 <UNK>에 가서 <UNK>을 먹었다. 특히 이전 단어들을 기반으로 다음단어를 예측하는 task에서 치명적 e.g. NaturalLanguageGeneration 어쨌든 모르는 단어지만, 알고있는 subword들을 통해 의미를 유출해볼 수 있음 🡪 e.g. 버카충ㅋㅋㅋㅋㅋㅋㅋㅋ 11. summary BPE 압축 알고리즘을 통해 통계적으로 더 작은 의미 단위(subword)로 분절 수행, BPE를 통해 OOV를 없앨 수 있으며, 이는 성능상 매우 큰 이점 작용 한국어의 경우 띄어쓰기가 제멋대로인 경우가 많아서 normalization 없이 바로 subword segmentation을 적용하는 것은 위험함 12. DETOKENIZATION 사람이 읽을 수 잇는 형태로 변환(index2word) 분절 복원 13. Question Q1. 구글의 Word Piece 기반 한국어 언어모델이 엑소브레인의 Word Piece 기반 한국어 언어모델보다 전체적으로 성능이 크게 떨어지는 것은 어떤 의미일까요? 여기서 유의해야 할 것은 언어모델(BERT)을 훈련시킨 원리는 동일하며, 토크나이저가 구성된 원리도 Word Piece 기반으로 동일하다는 점입니다.  Q2. 엑소브레인의 BERT에 두 가지 버전이 있는데, 이 중 한국어 전용 형태소분석기 토크나이저를 사용한 버전이 WordPiece 모델 토크나이저를 사용한 버전보다 대체로 성능이 좋다는 것의 시사점은 무엇일까요?  Q3. 정교한 형태소분석기를 활용한 모델의 성능이 더 좋을 수 있음에도 불구하고 현장에서 SentencePiece 같은 Subword 기반 토크나이저가 더욱 각광받는 이유는 무엇일까요?  생각해보기 |