# 12장 태깅 작업
# (Tagging Task)

PoS, NER, BIO 표현, BiLSTM-CRF 모델에 대해 알아봅시다!

2023년도 동계인턴 스터디 3주차
박성호

# 12-2 Part-of-Speech

단어의 품사가 명사, 동사, 형용사 인지를 결정

```
[('Pierre', 'NNP'), ('Vinken', 'NNP'), (',', ','), ('61', 'CD'), ('years', 'NNS'), ('old', 'JJ'), (',', ','), ('will', 'MD'), ('join', 'VB'), ('the', 'DT'), ('board', 'NN'), ('as', 'IN'), ('a', 'DT'), ('nonexecutive', 'JJ'), ('director', 'NN'), ('Nov.', 'NNP'), ('29', 'CD'), ('.', '.')]
```

케이

문장 데이터와 품사 데이터를 준비하고 작업을

케라스의 토크나이저를 통해 fit_on_texts()로 정수 인코딩(딕셔너리 만들기),
texts_to_sequences()로 정수 인코딩(정수로 변경)
pad_sequences()로 패딩 →

```
model = Sequential()
model.add(Embedding(vocab_size, embedding_dim, mask_zero=True))
model.add(Bidirectional(LSTM(hidden_units, return_sequences=True)))
model.add(TimeDistributed(Dense(tag_size, activation=('softmax'))))

model.compile(loss='sparse_categorical_crossentropy', optimizer=Adam(0.001), metrics=['accuracy'])
```

# 12-3 Named Entity Recognition

코퍼스로 부터 단어가 어떤 유형(사람, 장소, 단체 등)인지를 알아내는 작업

```
ner_sentence = ne_chunk(tokenized_sentence)
print(ner_sentence)
```

```
(S
  (PERSON James/NNP)
  is/VBZ
  working/VBG
  at/IN
  (ORGANIZATION Disney/NNP)
  in/IN
  (GPE London/NNP))
```

다.” → ‘유정’: 사람, ‘2018년’: 시간, ‘골드만 삭스’: 조직

)를 지원

해 품사 태깅(pos_tag) 선행

# 12-4 NER의 BIO 표현

개체명이 시작되는 Begin, 개체명의 내부 Inside, 개체명이 아닌 Outside

```
Peter NNP B-NP B-PER
Blackburn NNP I-NP I-PER
```

“애 메가박스 가자”

→애B(movie)니I(movie)포I(movie)터I(movie) 보O러O
메B(theater)가I(theater)박I(theater)스I(theater) 가O자O

- NER 데이터셋 CONLL2003는 [단어, 개체명 태깅] 형식으로 구성

  “EU rejects German call to boycott British lamb”

  → [NNP VBZ JJ NN TO VB JJ NN ] 품사 태깅
  [(B-NP) (B-VP) (B-NP I-NP) (B-VP I-VP) (B-NP I-NP)] 청크 태깅
  [B-ORG O B-MISC O O O B-MISC O ] 개체명 태깅

- 질문) ‘lamb’은 개체명 태깅에서 O?! 즉, Peter Blackburn은 하나의 개체명

# 오타..

2. 개체명 인식 데이터 이해하기 에서 'German에는 B-ORG라는 개체명 태깅이 붙습니다.' 요기 문장에 문맥상 German이 아니라 EU가 맞지 않을까요? - sungho park, 2023년 2월 1일 9:19 오후 , 대댓글 , 수정 , 삭제

오타 정정하였습니다. 감사합니다. - 유원준/안상준, 2023년 2월 1일 9:26 오후 , 대댓글

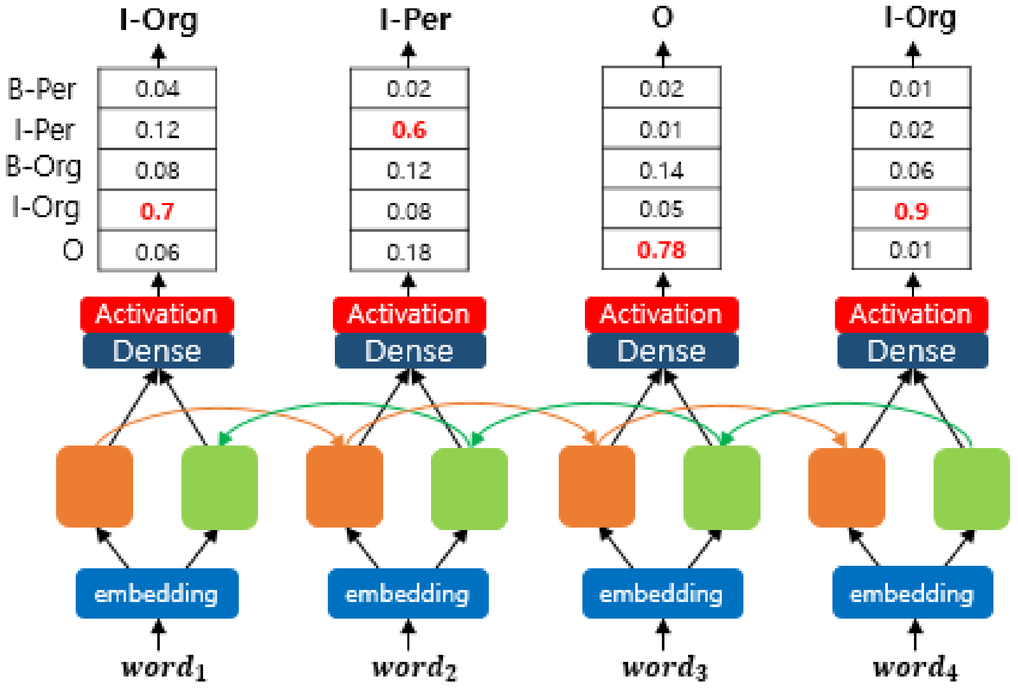
# NER 전처리 코드 일부

```
func = lambda temp: [(w, t) for w, t in zip(temp["Word"].values.tolist(), temp["Tag"].values.tolist())]
tagged_sentences=[t for t in data.groupby("Sentence #").apply(func)]
```

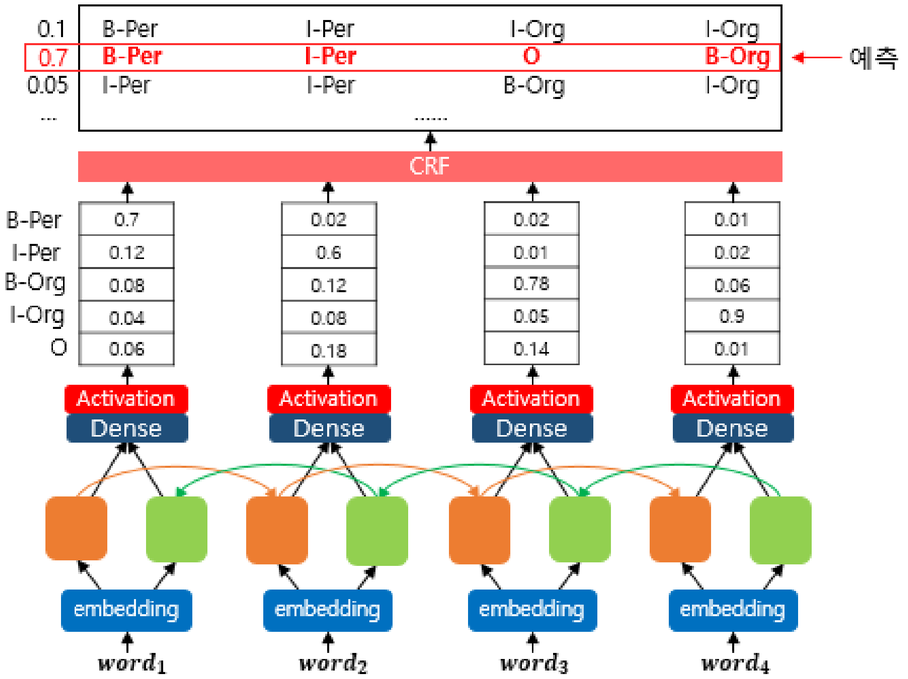
| | | | | |
|---|---|---|---|---|
| 1 | NaN | of | IN | O |
| 2 | NaN | demonstrators | NNS | O |
| 3 | NaN | have | VBP | O |
| 4 | NaN | marched | VBN | O |

# 12-6 Conditional Random Field + BiLSTM

CRF층을 추가해 개체명을 잘못 예측하는 상황을 방지할 수 있다




반!



# CRF+BiLSTM

- CRF층에서는 훈련데이터를 통해 다음 규칙(BIO 제약 조건)을 학습
  - 문장의 첫번째 단어에서는 I가 나오지 않는다.
  - O-I 패턴은 나오지 않는다
  - B-I-I 패턴에서 개체명은 일관성을 유지, (ex. B-Per 다음에 I-Org는 나오지 않음)
- keras-crf는 원-핫 인코딩 된 레이블은 지원하지 않는다.

# 13장 서브워드 토크나이저 (Subword Tokenizer)

Byte Pair Encoding에 대해 알아봅시다!

2023년도 동계인턴 스터디 3주차
박성호

# 13-1 Byte Pair Encoding

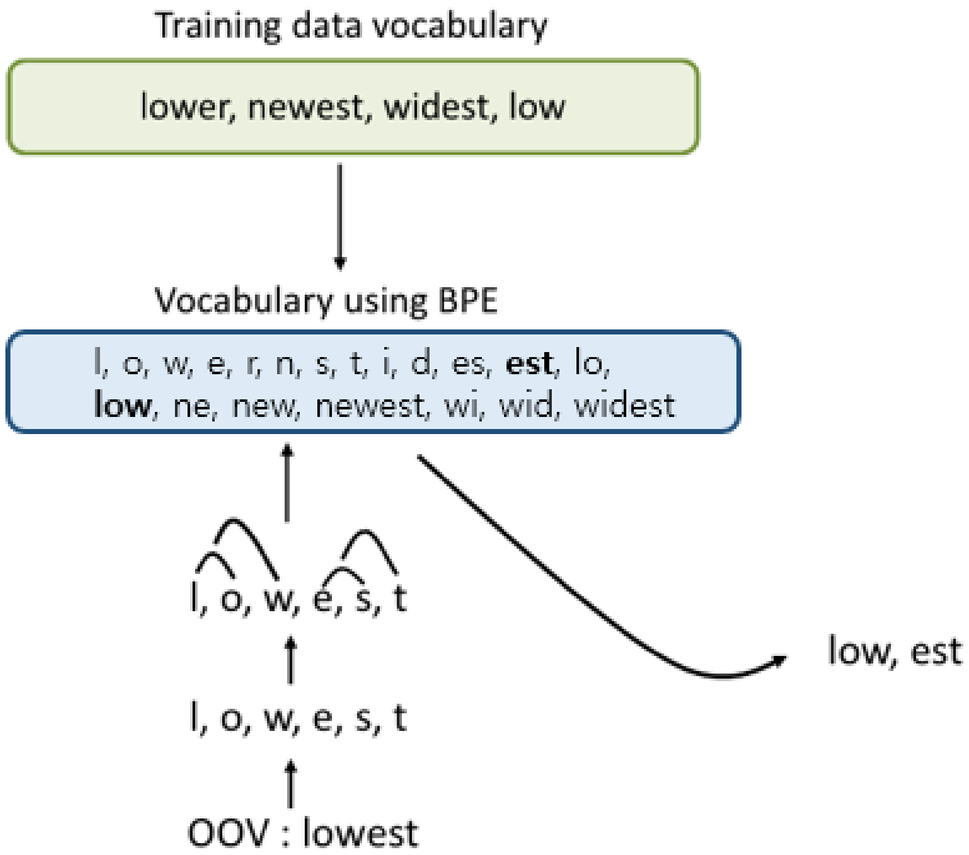
데이터 압축 알고리즘, 자연어 처리에서는 서브워드 분리 알고리즘

- 글자를 바이트로 보고, 글자의 쌍(=바이트의 쌍, BP)을 압축
  1. “aaabdaaabac” → 가장 많이 등장한 BP인 ‘aa’를 하나의 바이트 Z로 치환
  2. “ZabdZabac” →’ab’를 Y로 치환
  3. “ZYdZYac” → ‘ZY’를 X로 치환
  4. “XdXac” → 더 이상 병합할 BP가 없으므로 BPE 종료
- iteration은 사용자가 정의
- OOV 문제를 완화 할 수 있다.

# BPE 예시(1)

- (어떤 문서에서 ‘low’가 5번, ‘lower’가 2번, ‘newest’가 6번, ‘widest’가 3번 나왔을 때)
  {l o w: 5, l o w e r: 2, n e w e s t: 6, w i d e s t: 3}
  → ( l, o, w, e, r, n, s, t, i, d )
- Iteration 1: {l o w: 5, l o w e r: 2, n e w es t: 6, w i d es t: 3}
  → (l, o, w, e, r, n, s, t, i, d, es)
- Iteration 2: {l o w: 5, l o w e r: 2, n e w est: 6, w i d est: 3}
  → (l, o, w, e, r, n, s, t, i, d, es, est)
- Iteration 3: {lo w: 5, lo w e r: 2, n e w est: 6, w i d est: 3}
  → (l, o, w, e, r, n, s, t, i, d, es, est, lo)
- … Iteration 10: {low: 5, low e r: 2, newest: 6, widest: 3}
  → (l, o, w, e, r, n, s, t, i, d, es, est, lo, low, ne, new, newest, wi, wid, widest)

# BPE 예시(2)




l 들어온다면?

# WordPiece Tokenizer(1)

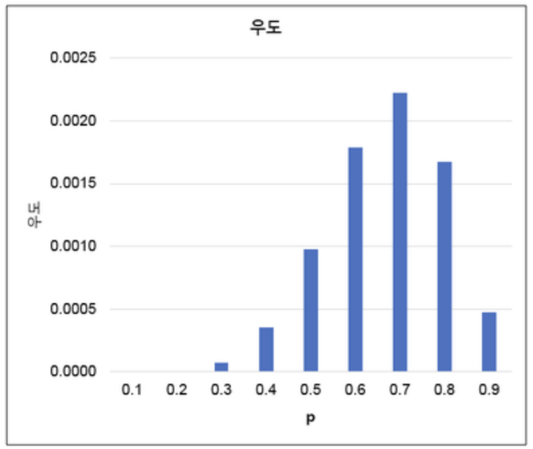
- BPE의 변형 알고리즘으로,빈도 수 대신 병합되었을 때 코퍼스의 가능도(Likelihood)를 가장 높이는 쌍을 병합
- 가능도(우도, Likehood): 데이터가 관측 됐을때, 특정 확률에 대한 믿음의 strength

$$\mathcal{L}(p|data) = \binom{n}{x} p^x (1-p)^{n-x} = P(data|p)$$

- 최대 우도법: 주어진 데이터를 가장 잘 표현하는 모수를 찾는 방법
- EX)동전을 10번 던졌을 때, 앞면이 7번 나온다

$$\mathcal{L} = \binom{10}{3} p^7 (1$$

| $n = 10$ | $x = 7$ |
|---|---|
| $p$ | 우도함수 값 |
| 0.1 | 0.000000073 |
| 0.2 | 0.000006554 |
| 0.3 | 0.000075014 |
| 0.4 | 0.000353894 |
| 0.5 | 0.000976563 |
| 0.6 | 0.001791590 |
| 0.7 | 0.002223566 |
| 0.8 | 0.001677722 |
| 0.9 | 0.000478297 |
| 합계 | 0.007583273 |



# WordPiece Tokenizer(2)

- WordPiece Tokenizer는 모든 단어의 맨 앞에 _를 붙이고,
  단어는 서브 워드로 통계에 기반하여 띄어쓰기로 분리
- “Jet makers feud over seat width with big orders at stake”
  →”_J et _makers _fe ud _over _seat _width _with _big _orders _at _stake”

# 13-2 SentencePiece

BPE + Unigram LM Tokenizer, 사전 토큰화 없이 단어 분리 토큰화 수행

- 언어에 종속되지 않는다는 특징
- “I didn't at all think of it this way.”
  - (서브워드 시퀀스로 변환)→ ['▁I', '▁didn', "'", 't', '▁at', '▁all', '▁think', '▁of', '▁it', '▁this', '▁way', ‘.’]
  - (정수 시퀀스로 변환) → [41, 623, 4950, 4926, 138, 169, 378, 30, 58, 73, 413, 4945]
- “진짜 최고의 영화입니다 ” 
  - (서브워드 시퀀스로 변환)→ ['▁진짜', '▁최고의', '▁영화입니다', ‘▁ㅋㅋ']

# 13-2 Huggingface Tokenizer

자연어 처리 스타트업 허깅페이스가 개발한 토크나이저

- 구글 BERT의 WordPiece Tokenizer를 직접 구현한 ‘BertWordPieceTokenizer’,
  오리지널 BPE인 ‘CharBPETokenizer’,
  BPE의 바이트 레벨 버전 ‘ByteLevelBPETokenizer’,
  SentencePiece와 호환되는 BPE 구현체 ‘SentencePieceBPETokenizer’ 등
- 네이버 리뷰 데이터를 BWPT로 학습해 얻은 3만개의 단어집합
  →[PAD], [UNK], [CLS], [SEP], [MASK],…, 말과, 말들이, 말라는, 말밖에는, 맘을
- “아 배고픈데 짜장면먹고싶다”를 토큰화 하면
  → [‘아', '배고', '##픈', '##데', '짜장면', '##먹고', '##싶다']


감사합니다.