스팬 기반 개체 추출을 위한 자질, 모델, 학습 방법 비교

이승우○
한국과학기술정보연구원 인공지능데이터연구단
swlee@kisti.re.kr

Comparing Features, Models and Training for Span-based Entity Extraction

Seungwoo Lee○
AI Data Research Center, Korea Institute of Science and Technology Information

요 약

개체 추출은 정보추출의 기초를 구성하는 태스크로, 관계 추출, 이벤트 추출 등 다양한 정보추출 태스크의 기반으로 중요하다. 최근에는 다중 레이블 개체와 중첩 개체를 다루기 위해 스팬기반의 개체추출이 주류로 연구되고 있다. 본 논문에서는 스팬을 표현하는 다양한 매핑과 자질들을 살펴보고 개체추출의 성능에 어떤 영향을 주는지를 분석하여 최적의 매핑 및 자질 조합을 제시하였다. 또한, 모델 구조에 있어서, 사전학습 언어모델(PLM) 위에 BiLSTM 블록의 추가 여부에 따른 성능 변화를 분석하고, 모델의 학습에 있어서, 미세조정(finetuing) 이전에 예열학습(warmup training)을 사용하는 것이 효과적인지를 실험을 통해 비교 분석하여 제시하였다.

주제어: 스팬 기반 개체 추출, 토큰열-스팬 매핑, 스팬 문맥 자질, 예열 학습

1. 서론

개체 추출(Entity Extraction)은 텍스트마이닝, 특히 정보추출의 기본이 되는 태스크로, 문장에서 고유 개체(Named Entity)나 일반 개체(Nominal Entity), 값 개체(Value Entity), 시간 개체(Time Entity) 등 다양한 유형의 개체의 범위를 찾아내고, 유형을 분류하는 태스크로 정의될 수 있다. 이 태스크를 해결하는 방법은 크게 세 가지로 나눠볼 수 있다. 첫째는 토큰열 레이블링 방식으로, 문장의 각 토큰을 개체유형별 BI(Begin, Inside)와 O(Outside) 레이블일 확률을 계산한 후, CRF(Conditional Random Field)를 통해 최적의 레이블 순서열을 결정함으로써 개체를 추출하는데[1][2], 비중첩 단일 레이블 개체 추출 태스크에서는 효과적이지만, 다중 레이블 또는 중첩 개체를 다루는 데에는 한계가 있다. 둘째는 기계독해(MRC) 방식으로, 문장과 개체유형을 질의하여 해당 유형에 속하는 개체의 범위(시작과 끝 위치)를 답하도록 하는 질의응답에 해당하는데[3], 추출해야 할 개체유형 수만큼 반복적으로 수행해야하기 때문에 비효율적인 단점이 있다. 셋째는 스팬(span) 방식으로, 문장에서 개체의 최대 길이 이내의 가능한 모든 토큰열을 개체후보로 나열하고 분류함으로써[4][5][6] 개체를 추출하기 때문에 다중 레이블 및 중첩 개체를 쉽게 다룰 수 있다는 장점이 있다. 개체의 최대 길이에 비례하여 개체후보의 수가 늘어나 계산량 및 메모리 사용량을 증가시키는 문제가 있지만, 토큰 단위로 스캔 경계를 먼저 예측한 후 예측된 스팬을 개체후보로 사용함으로써 완화될 수 있다[7]. 이와 같이, 다중 레이블 및 중첩 개체를 다룰 수 있다는 장점으로 최근에는 스팬 방식(스팬 나열식과 스팬 경계 예측식) 개체 추출이 주류로 연구되고 있다.

토큰열 레이블링 방식과 달리 스팬 방식에서는 하나 이상의 가변길이 토큰열이 스팬에 대응되기 때문이며 스팬을 어떻게 표현하고 인코딩하느냐가 성능에 중요한 요소가 된다. 따라서, 본 연구에서는 스팬을 구성하는 토큰열의 인코딩을 스팬의 인코딩으로 매핑하는 방식과 스팬의 좌우 및 내부 문맥 등의 기존에 알려진 자질들과 본 연구에서 새롭게 제안하는 자질들을 비교실험하여 성능 개선에 유효한 자질들을 제안하고자 한다. 또한, 모델 구조에 있어서도, 사전학습 언어모델(PLM) 위에 양방향장단기메모리(BiLSTM)[1][2]를 추가함에 따른 비교와 전체 모델을 학습(fine-tuning)하기 전에 PLM을 제외한 네트워크를 먼저 학습(warmup-training)하느냐에 따른 비교를 통해 개체 추출의 성능에 효과적인 방법을 제안하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구를 살펴보고, 3장에서는 기존에 알려진 자질들과 함께 새롭게 제안하는 자질들을 설명한다. 4장에서는 사용한 딥러닝 모델과 데이터 등의 실험환경, 그리고 실험 결과를 제시하며, 5장에서는 실험 결과를 바탕으로 결론을 짓는다.

2. 관련 연구

스팬 방식의 개체 추출을 위해 기존 연구에서 사용된, 스팬에 대응하는 토큰열의 인코딩을 스팬 인코딩으로 매핑하는 방식과 스팬의 좌우 및 내부 문맥 자질들에 대해 살펴본다.

먼저, 토큰열-스팬 인코딩 매핑 방식으로 first-last-cat이 다수의 연구들([4][5][8][9])에서 사용되었다. 이는 스팬을 구성하는 첫 번째 토큰 인코딩(first)과 마지막 토큰(last) 인코딩을 연결(concatenation)하여 스팬 인코딩을 구성하는 방법이다. [5]에서는 all-mean을 추가적으로 사용하였는데, 이는 스팬을 구성하는 모든 - first와 last를 포함 - 토큰의 인코딩을 평균(mean)하여 스팬 인코딩을 구성한 것이다. 서브-워드를 토큰 단위로 하는 사전학습 언어모델인 BERT[10]에서 서브-워드 인코딩을 워드 단위 인코딩으로 풀링(pooling)하기 위해 first 방식, 즉 워드의 첫 번째 서브-워드 인코딩을 그 워드의 인코딩으로 사용하는 방식을 취하였던 점과 달리, 개체 추출을 위해서는 개체 스팬을 구성하는 가능한 모든 토큰의 인코딩을 사용하는 방식이 주로 제안되어왔다.

다음으로, [11]에서는 스팬의 좌우 국소문맥(local context)을 자질로 활용하였다. 스팬의 왼쪽과 오른쪽 각각의 n개 토큰의 인코딩을 사용할 수 있는데, [11]에서는 좌우 1개씩의 토큰 인코딩이 사용되었다. [9]에서는 또한 스팬의 순차폭(sequential width)을 추가 자질로 사용하였는데, 이는 스팬에 포함된 토큰의 수를 가리키며, 개체의 길이(폭)에 따른 확률적 분포를 반영하기 위한 자질이다.

본 연구에서는 앞서 살펴본 알려진 자질들과 함께, 추가적으로 가능한 토큰열-스팬 인코딩 매핑 방식과 스팬의 문맥 자질들을 살펴보고, 이 자질들을 실험을 통해 비교하여 최적의 자질 조합을 제시하고자 한다. 또한, 이러한 자질들을 BiLSTM[1][2] 네트워크 사용 여부와 미세조정(fine-tuning)전 예열학습(warmup-training) 사용 여부와 결합하여 개체 추출의 성능에 유리한 조합을 제시하고자 한다.

3. 제안 방법

스팬 방식의 개체 추출에서는 최대 길이(L)이내의 모든 가능한 토큰열(스팬)을 개체 후보로 간주하고 스팬에 대한 인코딩 표현을 구성하여 softmax (다중 분류) 혹은 sigmoid (이진 분류)를 통해 분류하기 때문에, 스팬 표현을 어떻게 구성하는지가 개체 추출의 성능을 좌우하는 중요한 요소이다. 스팬 표현은 토큰열-스팬 매핑과 스팬 문맥 자질로 나눠볼 수 있다.

3.1 토큰열-스팬 매핑

서브-워드를 토큰 단위로 하는 BERT의 인코딩을 워드 단위로 풀링(pooling)하기 위해 first 방식이 사용되었는데[10][12], 이후 다른 연구[13]에서는 all-mean 방식이 보다 효과적임이 제시된 바 있다. 특히, 스팬 표현에 있어서 first 방식은 시작 토큰을 공유하는 길이가 다른 여러 스팬 후보들간을 구분하지 못하는 단점을 야기한다. 따라서, 적어도 스팬의 시작과 끝 토큰을 포함하도록 구성할 필요가 있으며, 다음과 같이 세 가지의 스팬 경계 매핑 방식이 가능하다. 또한, 스팬의 경계(시작과 끝)을 제외한 스팬 내부 토큰들을 매핑하는 두 가지의 스팬 내부 매핑 방식이 가능하다.

- 스팬경계: first-last-cat, first-last-mean, first-last-max, first-last-sum
- 스팬내부: inner-mean, inner-max, inner-sum

스팬경계 매핑 중 first-last-cat은 기존 연구들[4][5][8][9]에서 주로 사용되었지만, 다른 두 가지 매핑과의 비교가 제시되지는 않았다. first-last-mean은 시작 토큰과 끝 토큰의 차원별 평균을 가리키며, 마찬가지로, first-last-max는 최대치를 가리킨다. first-last-sum 매핑 또한 가능할 수 있지만, 합계(sum)는 길이가 긴 스팬일수록 분산을 증가시키는 경향이 있는데, 이를 완화하기 위해, 스팬 내의 최대 및 최소값으로 제한(clipping)을 두도록 하였다. 길이가 1인 스팬의 경우, 시작 토큰과 끝 토큰은 동일하다.

스팬내부 매핑으로, [5]에서는 스팬의 시작과 끝 토큰을 포함하여 스팬 내부의 모든 토큰 인코딩의 평균을 취하는 all-mean 방식을 사용하였는데, 본 연구에서는 스팬 시작과 끝 토큰은 스팬경계 매핑에서 이미 다뤄지기 때문에 시작과 끝 토큰을 제외한 순수 내부 토큰들만 사용하여 평균(inner-mean)하거나 최대치(inner-max)로 매핑하였다. 내부를 모두 합하는 방식(inner-sum)도 가능한데, 스팬경계에서와 마찬가지로 길이가 긴 스팬의 분산 증가를 완화시키기 위해 최대-최소값 제한을 두도록 하였다. 길이가 1~2인 스팬의 경우 순수 내부 토큰이 없기 때문에 이 경우에는 영(zero) 벡터를 사용하였다.

3.2 스팬 문맥 자질

스팬의 내부 및 외부 문맥 자질들로서, 스팬의 개체 유형 분류에 도움을 줄 수 있는 정보로는 다음과 같이, 스팬의 폭과 스팬LCA, 스팬국소문맥, 문장문맥이 가능하다.

- 스팬폭: 순차폭(seq. width), 구문폭(syn. width)
- 스팬LCA: 스팬을 구성하는 모든 토큰들의 LCA (Least Common Ancestor)
- 국소문맥: 좌우 n-gram 토큰열
- 문장문맥: cls, all-mean, all-max

스팬 폭은 다시 스팬을 구성하는 토큰의 수를 가리키는 순차폭(sequential width)과 구문트리에서 스팬의 각 토큰간 구문거리의 최대치를 가리키는 구문폭(syntactic width)으로 나눠볼 수 있다. 두 토큰 i와 j간의 구문거리는 $d_i + d_j - 2 \times d_{LCA_{ij}}$로 계산될 수 있는데, d_i는 토큰i의 루트로부터의 깊이, LCA_{ij}는 토큰i와 토큰j의 최소공통조상(Least Common Ancestor)을 가리킨다. 스팬폭은 유형별로 개체가 갖는 토큰의 수나 구문적 거리에 대한 확률적 분포가 다를 수 있음을 가정한 것이다. 스팬폭은 정수값이며, 작은값 범위보다는 큰값 범위에서 편차를 줄이기 위해 로그(log)를 취한 후, 무작위 초기화된 임베딩 벡터로 매핑되도록 하였다.

스팬LCA는 스팬을 구성하는 모든 토큰들의 구문적 최소공통조상(Least Common Ancestor)인 토큰의 인코딩을 자질로 사용함을 가리킨다. 스팬 내부의 토큰 중 하나가

스팬LCA가 될 수 있으며, 이 경우에는 스팬LCA가 스팬내 토큰들의 구문적 head역할을 하는 것이기 때문에, 스팬 유형 판단에 의미있는 자질이 될 수 있을 것으로 판단된다. 스팬LCA가 스팬 외부에 있는 토큰일 경우는 스팬이 수식하는 또는 수식을 받는 관계에 있다고 볼 수 있기 때문에, 이 경우에도 스팬LCA는 스팬의 유형 구분에 유효한 자질이 될 수 있다고 판단된다. [14]에서는 트리거와 논항 간의 부분 구문트리를 얻기 위해 LCA를 사용한 바 있지만, 스팬 표현에서의 사용은 처음이다.

스팬국소문맥(local context)는 스팬의 좌우에 위치하는 각 n개의 토큰열의 인코딩을 가리며, 스팬의 경계에 인접하여 스팬의 유형 분류에 유효한 자질이 될 수 있다고 판단된다.

이상의 자질들은 모두 스팬의 내부 및 스팬에 인접한 토큰들로부터 얻어지기 때문에, 스팬이 포함된 문장의 전체적인 의미와 맥락을 반영하지는 못한다. 이를 보완하기 위해, 문장문맥은 스팬이 포함된 문장의 전체적인 의미 자질을 반영하기 위한 것으로, cls, all-mean, all-max 등 세 가지 방식이 가능하다. cls는 BERT에서 제공하는 출력의 맨 첫 번째 토큰 인코딩으로[10], 이는 BERT의 사전학습에서 인접한 두 문장의 연속성 여부를 분류하는 용도로 학습된 것이기 때문에 문장의 전체적인 의미를 잘 반영하고 있는 벡터로 간주된다. 나머지 세 가지는 각각 문장의 모든 토큰 인코딩을 평균(all-mean), 최대치(all-max)한 것을 가리킨다. all-sum도 가능하지만, 길이가 긴 문장에 대해 합계를 할 경우, 최대-최소값으로 제한을 하더라도 분산을 상당히 증가시켜 학습을 저해하는 경향이 커서 배제하였다.

토큰열-스팬 매핑과 스팬 문맥 자질 중에서 평균(*-mean)과 최대치(*-max)를 구하는 데에 주의가 요구된다. 유효 문장 길이와 유효 스팬 길이는 가변적이기 때문에 패딩(padding)을 포함하고 있는데, 이 패딩이 평균과 최대치를 왜곡시킬 수 있기 때문이다. 패딩을 제외하기 위해 마스크(mask)를 사용해야 하며, 평균 계산에서는 합계(sum)를 마스크의 합으로 나누는 방식으로 계산되어야 하며, 최대치 계산에서는 패딩에 해당하는 부분을 충분히 작은 음수(eg, -1e-9)로 치환한 후 최대치 연산을 함으로써, 패딩에 의해 모든 음의 값이 0으로 치환되는 왜곡을 피해야 한다.

이상의 매핑 및 자질들이 스팬 방식의 개체추출에서 효과적인지 여부는 4장에서 실험 결과를 통해 제시하였다.

3.3 예열 학습

최근의 텍스트 마이닝 심층신경망은 대부분 사전학습 언어모델(PLM)을 기반으로 하며, 응용 태스크에 맞게 미세조정(finetuning)하는 방식을 취한다. PLM 위에 태스크에 따른 분류를 수행할 몇 개의 디코딩(decoding) 계층을 추가하게 되는데, 이 디코딩 계층은 무작위로 초기화되는 것이 일반적이다. 반면, PLM은 응용 태스크와는 다르지만, MLM (Masked Language Model)과 NSP (Next Sentence Prediction) 태스크로 충분히 학습을 거친 상태이다. PLM과 디코딩 네트워크를 포함한 전체 모델을 응용 태스크 데이터로 미세조정(finetuning) 학습할 수 있지만, PLM과 디코딩 네트워크 간의 가중치 비일관성으로 인해, 학습 초기에 불안정한 경향을 보일 수 있으며, 이는 최종 학습 효과를 저해할 수 있다. 이런 점을 완화하기 위해, 미세조정 전에 작은 에포크 동안의 예열학습(warmup training)을 수행하는 것이 효과적일 수 있다. 4장에서 실험을 통해, 예열학습의 여부에 따른 개체추출 성능 차이를 제시하였다.

4. 실험

4.1 실험 환경

스팬 방식의 개체추출을 실험하기 위한 데이터셋으로 CoNLL2003 영어 데이터셋[15]을 사용하였다. 학습-개발-테스트용으로 각각 946개, 216개, 231개의 문서로 구성되어 있으며, PER(사람명)과 LOC(장소명), ORG(기관명), MISC(기타) 등 4개의 고유 개체가 태깅되어 있다. 학습용으로 6,600개의 PER, 7,140개의 LOC, 6,321개의 ORG, 3,438개의 MISC 등 개체 유형별로 충분한 수의 개체가 태깅되어 있어서, 심층신경망을 학습하고 자질을 평가하기에 적합하다고 판단하였다. 학습용 데이터로 모델을 학습하고 개발용 데이터셋에 대한 micro F1을 기준으로 모델을 선정하여 테스트용 데이테셋에 대한 micro F1 성능을 측정하였다.

문장의 토큰 인코딩을 위한 사전학습 언어모델로는 BERT-base-cased를 사용하였는데, 서브-워드 단위를 워드 단위로 풀링(pooling)하기 위해 예비실험에서 가장 좋은 결과를 보여준 first-last-max 방식을 선택하였다. BERT가 어텐션 기반으로 문장 내 토큰들 간의 관계를 인코딩해주지만, 토큰들간의 순차적인 관계를 추가로 학습하기 위해 BiLSTM블록 (BiLSTM - Add&Norm - FFN - Add&Norm 네트워크)을 BERT 위에 추가하였다. 이는 트랜스포머 인코더/디코더 블록[16]의 구조에 착안한 것이다. 이웃한 문장간의 문맥을 공유할 수 있도록, 대상 문장을 중심에 두고 최대 입력 길이 (128 토큰 또는 168 서브-워드)이내에서 이전 및 이후 문장들을 좌우에 함께 입력하여 인코딩되도록 하였다. BiLSTM블록을 거친 후, 대상 문장만 골라내어 최대 길이 10 토큰 이내의 가능한 스팬 표현을 생성하고 FFN - Softmax를 통해 개체유형을 분류하였다. 스팬 자질을 추출에 필요한 의존구문트리는 spaCy[1] 라이브러리가 사용되었다. 국소문맥으로는 좌우 각각 1개(n=1)의 토큰을 사용하였고, 스팬폭 임베딩으로는 20개의 20차원 벡터가 사용되었다. 외부 자원으로는 one-hot으로 인코딩된 개체명사전(nedict)[17]이 사용되었다.

모델의 학습 동안 사용된 주요 하이퍼파라미터는 다음과 같다.

- 최적화 알고리즘: AdamW with weight decay(=1e-4)
- 학습률: warmup-lr=(1e-3, 1e-4), ft-lr=(5e-4,

1) https://spacy.io/

0), 10% warmup 및 선형감소(LinearDecay)스케줄링
- 배치크기: 32
- 에포크: 3 (warmup) + 10 (finetune) 또는 13 (finetune-only)
- dropout rate: 0.1

4.2 실험 결과

앞서 설명한 바와 같이, CoNLL2003 학습용 데이터를 사용하여 스팬 기반 개체추출 모델을 학습한 후 개발용 데이터에 대한 micro F1을 기준으로 모델을 선정하고 테스트용 데이터에 대해 micro F1을 측정하였다. 각각의 모델 구성 및 자질 조합별로 5회 반복 실행 후 평균과 표준편차를 계산하였다.

먼저, 토큰열-스팬 매핑과 스팬 문맥 자질들이 개체추출에 효과적인지를 살펴보았다. 표 1은 BiLSTM블록과 예열학습을 사용하는 환경에서, 각 자질들의 조합에 따른 개체추출 성능(F1)을 측정한 결과이다. 먼저, 토큰열-스팬 매핑의 경우, 스팬 경계와 스팬 내부 매핑을 모두 사용한다면, 세부적인 매핑은 어느 방식을 쓰더라도 성능에 거의 차이를 보이지 않았다. 나머지 자질 조합을 위해, 편의상 first-last-cat + inner-mean을 선택하여 기본 자질로 사용하였다. 이 기본 자질에 국소문맥(lcvec)과 개체명 사전(nedict)을 추가할 경우, 개발용에서는 성능 차이가 미미하지만, 테스트용에서는 0.3%p 이상의 개선을 보였다. 추가적으로, 스팬LCA, 스팬폭, 문장문맥(sentpool) 자질들을 각각 결합한 실험에서는 구문폭(synwidth)가 개발용 F1을, 문장문맥-cls가 테스트용 F1 성능을 개선하는 것으로 나타났다. 반면, 문장문맥 all-max와 all-mean은 성능을 일부 저해하는 것으로 나타났다.

표 1 토큰열-스팬 매핑과 스팬 문맥 자질들의 조합에 따른 개체추출 성능 비교 (BiLSTM블록 및 예열학습 사용)

ID	자질조합	개발용		테스트용	
		F1	Diff	F1	Diff
(0)	first-last-cat + inner-mean	96.61	-	92.88	-
	first-last-cat + inner-max	96.59	-0.01	92.86	-0.02
	first-last-cat + inner-sum	96.61	0.01	92.90	0.03
	first-last-mean + inner-mean	96.61	0.01	92.79	-0.08
	first-last-max + inner-mean	96.63	0.02	92.86	-0.02
	first-last-sum + inner-mean	96.57	-0.04	92.94	0.06
(1)	(0) + lcvec	96.64	0.03	93.19	0.31
(2)	(1) + nedict	96.61	0.00	93.27	0.39
	(2) + spanlca	96.65	0.05	93.25	0.37
	(2) + seqwidth	96.60	0.00	93.14	0.27
	(2) + synwidth	**96.67**	**0.06**	93.31	0.43
	(2) + sentpool-cls	96.61	0.01	**93.34**	**0.46**
	(2) + sentpool-all-mean	96.66	0.05	92.93	0.05
	(2) + sentpool-all-max	96.47	-0.14	93.02	0.14

표 2 주요 자질 조합별 BiLSTM 블록 사용 여부에 따른 개체추출 성능 비교 (예열학습 사용)

ID	Bi-LSTM	자질조합	개발용		테스트용	
			F1	Diff	F1	Diff
(0)		first-last-cat + inner-mean	96.53	-	92.53	-
(1)	✓	first-last-cat + inner-mean	96.61	0.08	92.88	0.35
(2)		(0) + lcvec	96.61	-	93.05	-
(3)	✓	(1) + lcvec	**96.64**	0.03	93.19	0.13
		(2) + nedict	96.53	-	93.13	-
	✓	(3) + nedict	96.61	0.08	**93.27**	0.14

다음으로, 주요 자질 조합별로 BiLSTM 블록을 사용할 경우와 그렇지 않을 경우의 개체추출 성능을 비교하였다. 표 2에서 보는 바와 같이, 세 가지 주요 자질을 순차적으로 추가하는 상황에서 BiLSTM 블록을 사용할 경우가 항상 더 나은 성능을 보여주었다. 개발용 보다는 테스트용 데이터에서 성능 향상 폭이 컸으며, 0.13~0.35%p의 성능 개선을 보였다. 이는 개체추출 태스크에 있어서 BiLSTM 블록을 추가로 사용하는 것이 성능개선에 유리함을 확인시켜준다.

마지막으로 주요 자질 조합과 모델구조별 예열학습 사용 여부에 따라 개체추출의 성능 변화가 있는지를 분석하였다. 표 3은 BiLSTM블록 사용 여부와 주요 자질 조합에 대해 예열학습을 사용하지 않을 경우 대비 사용할 경우의 성능 차이(Diff)를 보여준다. 모두 여섯 가지 서로 다른 조합에서 예열학습을 사용할 경우가 항상 성능을 개선시키는 것으로 나타났으며, 개발용에서는 0.4~0.6%p, 테스트용에서는 0.13~0.88%p만큼 F1 성능이 개선되는 것으로 분석되었다. 이를 통해, 예열학습을 사용하는 것이 개체추출 성능 개선에 상당히 효과적임을 알 수 있다.

표 3 주요 자질 조합 및 모델구조별 예열학습 사용 여부에 따른 개체추출 성능 비교

ID	Bi-LSTM	예열	자질조합	개발용		테스트용	
				F1	Diff	F1	Diff
(0)			first-last-cat + inner-mean	96.02	-	92.40	-
(1)		✓	first-last-cat + inner-mean	96.53	0.51	92.53	0.13
(2)			(0) + lcvec	96.09	-	92.33	-
(3)		✓	(1) + lcvec	96.61	0.51	93.05	0.73
			(2) + nedict	95.98	-	92.24	-
		✓	(3) + nedict	96.53	0.54	93.13	0.88
(4)	✓		first-last-cat + inner-mean	95.99	-	92.56	-
(5)	✓	✓	first-last-cat + inner-mean	96.61	0.62	92.88	0.32
(6)	✓		(4) + lcvec	96.21	-	92.58	-
(7)	✓	✓	(5) + lcvec	96.64	0.43	93.19	0.61
	✓		(6) + nedict	96.21	-	92.60	-
	✓	✓	(7) + nedict	96.61	0.40	93.27	0.67

5. 결론

본 연구에서는 스팬 기반의 개체추출 딥러닝 모델에서 사용될 수 있는 다양한 스팬 표현 자질들을 살펴보고 각각의 자질들이 실제 개체추출 성능에 어떤 효과를 보여주는지를 실험을 통해 확인하고 분석하였다. 토큰열-스팬 매핑에 있어서는 스팬경계 매핑과 스팬내부 매핑을 각각 사용하는 것이 효과적이며, 세부적인 매핑 방법의 차이는 성능에 영향을 주지 않는 것으로 분석되었다. 스팬 문맥 자질들 중에서는 국소문맥(lcvec)과 구문폭

(synwidth), cls 문장문맥이 스팬 매핑 자질과 함께 사용될 때 성능 향상에 효과적임이 확인되었다.

모델 구조에 있어서는, 사전학습 언어모델(PLM) 위에 BiLSTM 블록을 추가로 사용하는 것이 테스트 성능에서 0.13~0.35%p의 성능 개선 효과를 보여주었고, 모델 학습에 있어서는, 미세조정(finetuning) 전에 추가된 네트워크에 대한 예열학습(warmup training)을 수행하는 것이 0.13~0.88%p의 테스트 성능 향상을 보여주어 상당히 효과적임을 확인할 수 있었다.

본 연구의 개체추출 성능 93.34 F1은 비슷한 규모의 언어모델과 자원을 사용하는 기존 연구들과 비슷하거나 견줄 만한 수준이지만, CoNLL2003 데이터에 대한 SoTA 성능과는 1%p 남짓 차이가 있다. 보다 개선된 대규모 언어모델과의 결합, 외부 학습 데이터 보강, 외부 지식 자원의 활용 등 다양한 방안을 결합하여 지속적으로 성능을 개선할 예정이며, 또한 길이가 긴 스팬을 처리하는 데 있어서, 메모리 사용량과 계산량에 부하를 발생시키는 스팬 나열 방식을 보다 효과적인 스팬 경계 예측 방식으로 확장할 예정이다.

Acknowledgement

본 연구는 한국과학기술정보연구원의 'Data/AI 기반 문제해결 체계 구축(K-23-L04-C05)' 사업으로부터 지원받아 수행된 연구임.

참고문헌

[1] Z. Huang, W. Xu and K. Yu, Bidirectional LSTM-CRF Models for Sequence Tagging, arXiv:1508.01991, 2015.

[2] X. Ma and E. Hovy, End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF, arXiv:160.01354v5, 2016.

[3] X. Li, J. Feng, Y. Meng, Q. Han, F. Wu and J. Li, A Unified MRC Framework for Named Entigy Recognition, Proceedings of the 58th Annual Meeting of the ACL, pp.5849-5859, 2020.

[4] Z. Zhong and D. Chen, A Frustratingly Easy Approach for Entity and Relation Extraction, Proceedings of the 2021 Conference of the NAACL: HLT, pp. 50-61, 2021.

[5] H. Trieu, T.T. Tran, K.N.A. Duong, A. Nguyen, M. Miwa and S. Ananiadou, DeepEventMine: end-to-end neural nested event extraction from biomedical texts, Bioinformatics, vol.36, issue 19, pp.4910-4917, 2020.

[6] W. Gu, B. Zheng, Y. Chen, T. Chen, B.V. Durme, An Empirical Study on Finding Spans, Proceedings of the 2022 Conference on EMNLP, pp. 3976-3983, 2022.

[7] Y. Chen, L. Wu, Q. Zheng, R. Huang, J. Liu, L. Deng, J. Yu, Y. Qing, B. Dong and P. Chen, A Boundary Regression Model for Nested Named Entity Recognition, Cognitive Computation, 15, pp. 534-551, 2023.

[8] D. Ye, Y. Lin and M. Sun, Pack Together: Entity and Relation Extraction with Levitated Marker, arXiv:2109.06067v1, 2021.

[9] D. Wadden, U. Wennberg, Y. Luan and H. Hajishirzi, Entity, Relation, and Event Extraction with Contextualized Span Representations, arXiv:1909.03546v2, 2019.

[10] J. Devlin, M.W. Chang, K. Lee and K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805v2, 2018.

[11] T.H. Nguyen, K. Cho and R. Grishman, Joint Event Extraction via Recurrent Neural Networks, Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp.300-309, 2016.

[12] Y. Lin, H. Ji, F. Huang and L. Wu. A Joint Neural Model for Information Extraction with Global Features, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp.7999-8009, 2020.

[13] T.M. Nguyen and T.H. Nguyen, One for All: Neural Joint Modeling of Entities and Events, The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19), pp.6851-6858, 2019.

[14] M. Lee, L.-K. Soon, E.G. Siew, Effective Use of Graph Convolution Network and Contextual Sub-Tree for Commodity News Event Extraction, Proceedings of the Third Workshop on Economics and Natural Language Processing, pp.69-81, 2021.

[15] E.F.T.K. Sang and F.D. Meulder, Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition, arXiv:cs/0306050v1, 2003.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention Is All You Need, arXiv:1706.03762, 2017.

[17] C. Ronran, S. Lee and H.J. Jang, Delayed Combination of Feature Embedding in Bidirectional LSTM CRF for NER, Applied Sciences, 10(21), 7557, https://doi.org/10.3390/app10217557, 2020.