Word Sense Disambiguation (WSD)

이하 '본 논문', '논문'으로 지칭하는 논문은 다음과 같다. Huang, Luyao & Chi, Sun & Qiu, Xipeng & Huang, Xuanjing. (2019). GlossBERT: BERT for Word Sense Disambiguation with Gloss Knowledge.

Explain details of the task and dataset

Word Sense Disambiguation (이하 WSD)는 주어진 맥락에서 중의적인 단어가 어떠한 의미로 사용되었는지를 찾는 과제이다. 이 과제를 더욱 잘 수행하기 위해서, 우리는 다음 두 가지 지점을 살펴보기로 하였다.

1) 품사 정보

논문에서 든 예시를 통해 품사 정보를 활용하는 것이 WSD에 도움이 될 것으로 예상하였다. 논문은 예시 문장에서 사용된 단어 'research'가 'research'의 네 가지 뜻 중 하나에 해당한다고 하였는데, 이 때 소개된 'research'의 네 가지 뜻 중 두 개는 명사였고 두 개는 동사였다. 이에 품사 정보를 입력하는 경우 WSD에 도움이 될 것으로 생각해 보았다.

품사 정보를 추가하기 위해 다음의 방법을 사용하였다. 데이터는 github에 공개된 semcor_train_sent_cls_ws.csv를 변형하여 사용하였다.[1] 이 csv 파일은 GlossBERT를 고안한 연구자들이 SemCor3.0 데이터를 가공한 뒤 공개한 파일이다. 여기에 nltk의 wordnet 라이브러리를 활용하여 품사 정보를 추가하였다.[2] 논문에서 WordNet을 사용하였으므로 csv 파일에 제시된 sense_key와 nltk wordnet의 pos 메서드를 이용하여 품사 정보를 얻을 수 있었다.

2) 유의어 정보

유의어 정보를 활용하는 것도 WSD에 도움이 될 것으로 예상하였다. 비록 사용한 목적과 방법이 상이하기는 하나, 유의어, 상위어, 하위어의 관계를 활용하여 보다 적은 데이터로 효과적인 WSD를 가능하게 한 선행연구에서 아이디어를 얻었다.[3] 앞서 언급한 nltk의 wordnet 라이브러리를 활용하는 경우 sense_key에 대응되는 유의어를 얻어낼 수 있었다.

---

[1] https://github.com/HSLCY/GlossBERT

[2] https://www.nltk.org/howto/wordnet.html

[3] Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2019. Sense Vocabulary Compression through the Semantic Knowledge of WordNet for Neural Word Sense Disambiguation. In *Proceedings of the 10th Global Wordnet Conference*, pages 108–117, Wroclaw, Poland. Global Wordnet Association.

Give a summarization of the selected paper

본 논문은 WSD를 위해 gloss 정보를 어떻게 활용할 지 고민하였다. 기존 선행연구에서는 gloss 정보가 성능을 유의미하게 개선시키지 못했다고 하는데, 본 논문에서는 gloss 정보가 WSD 성능을 최소 5.8%p, 최대 10.9%p 향상시켰다.

본 논문은 GlossBERT를 제안하였다. GlossBERT는 context-gloss쌍을 받은 뒤, 주어진 gloss가 타겟 단어의 gloss에 해당하는지 여부를 판단하여 yes 또는 no로 분류한다. GlossBERT를 학습시킨 방법은 다음과 같다. 논문에서는 gloss 정보를 활용하는 세 가지 방법을 비교하였다. 첫 번째 방법은 Token-CLS 방법이다. 이는 타겟 단어 토큰의 마지막 hidden state를 활용한다. 두 번째 방법은 Sent-CLS 방법이다. 이는 단어 임베딩이 아닌 문장 임베딩을 사용한다. [CLS] 토큰에 문장 정보가 반영되어 있다고 보고 [CLS] 토큰의 마지막 hidden state를 활용한다. 세 번째 방법은 Sent-CLS-WS 방법이다. 이는 문장 임베딩을 사용하되 타켓 단어의 위치에 대해 weak supervision을 한 것이다. 세 번째 방법은 앞서 언급한 두 방법을 절충 및 혼합한 것이라고 볼 수 있으며, 대부분의 경우 가장 성능이 높았다. 이렇게 만든 GlossBERT는 기존 BERT 모델에 비해 WSD 성능이 높았다.

Explain why you picked the model for the task, based on which properties of task and model

WSD를 위해 GlossBERT를 활용한 이유는 GlossBERT의 두 가지 장점 때문이다. 이 장점들은 논문에도 언급되어 있다. 첫째, WSD를 이진분류 task로 전환하여 데이터를 불린 효과를 얻을 수 있었기 때문이다. 둘째, gloss 정보를 더 잘 활용하게 되었기 때문이다. 이는 기존 BERT 모델에 비해 성능이 높다는 점을 통해 확인할 수 있는 부분이다.

Analyze and Discuss: Did the different model that you picked or edited work better than the original method? If or if not, why?

1) 품사 정보

결과를 보고하기 이전에 품사 정보와 성능의 관계에 대해 생각해본다. (1) 품사 정보가 성능을 향상시킨다면, 이는 단어의 의미(word sense)가 품사와 관련이 되어있음에도 불구하고 기존의 모델이 이 정보를 이용하지 못했기 때문으로 해석할 수 있다. (2) 한편 품사 정보가 성능을 향상시키지 못한다면 기존의 모델이 이미 품사 정보를 활용하고 있었던 것으로 해석할 수 있다. (3) 품사 정보가 오히려 성능을 떨어뜨린다면, 그 원인으로 다음을 생각해볼 수 있다. GlossBERT가 만들어진 과정과 평가과정에는 품사 정보가 포함되지 않았기 때문이다. 품사 정보 이전 학습, 품사 정보를 포함한 학습, 평가 사이에 괴리가 발생하여 기존의 학습이 왜곡되었을 가능성이 있다.

2) 유의어 정보

결과를 보고하기 이전에 유의어 정보와 성능의 관계에 대해 생각해본다. (1) 유의어 정보가 성능을 향상시킨다면, 모델이 유의어를 통해 단어의 의미를 보다 잘 학습한 것이다. (2) 유의어 정보가 성능을 향상시키지 못한다면 유의어를 통해 추가적인 정보를 얻지 못한 것이다. 유의어의 정보가 이미 gloss에 포함되어 있기 때문이라고 생각해볼 수 있다. (3) 유의어 정보가 성능을 오히려 하락시킨다면 이는 타겟 단어와 유의어 사이에 존재하는 의미차이 때문일 것으로 생각한다. 혹은 앞서 언급했던 이전학습, 평가와의 괴리 때문일 가능성이 있다.

3) 결과보고

논문에서 사용한 평가데이터와 동일한 데이터와 동일한 평가방식을 사용하여 위 모델을 평가하였다. 동일한 데이터라 함은 품사 정보나 유의어 정보를 반영하지 않은, github에 공개된 데이터를 그대로 사용하였음을 의미한다.[4] 한편 동일한 평가방식이라 함은 한 문장에서 사용된 다의어의 의미를 맞추었는지 여부를 이진분류로 1회 판단하였음을 의미한다.

평가방식에 대해 조금 더 상술하자면, 모델 학습시에는 하나의 타겟문장이 그 문장에서 사용된 특정 다의어의 여러 의미와 함께 짝지어져 있는 형식의 데이터를 활용하였다. 즉, 어떤 한 문장이 하나의 다의어를 포함하고 있고 그 다의어가 N개의 의미를 갖고 있다면, 한 문장에 대해 N개의 데이터가 주어지는 것이다.

그러나 평가 시에는 [타겟문장 - 해당 문장에서 사용한 다의어의 **정답** 의미]와 모델을 통해 구한 [타겟문장 - 해당 문장에서 사용했다고 추측한 다의어의 **예상** 의미]의 비교를 통해 한 문장에 대해 1회만 평가를 실시하였다.

결과는 다음과 같다. 논문에 보고된 GlossBERT(Sent-CLS-WS)의 F1 score는 72.5%였다. 그러나 품사와 유의어 정보를 학습시킨 뒤 구한 F1 score는 71.9%였다. 이는 기존에 비해 0.6%p 하락한 수치이다.

학습이 충분히 이루어지지 않았을 가능성을 고려하여 epoch를 더 돌리는 방법을 시도하였다. 그러나 추가적인 epoch의 32,500 step까지 성능에 유의미한 변화는 나타나지 않았다. 이후 추가적인 학습은 컴퓨팅 용량의 제한으로 인해 진행하지 못하였다. 이하 학습이 완전히 이루어졌다고 보고 위와 같은 결과가 나타난 이유에 대해 생각해본다.

첫째, 불연속성을 들 수 있다. GlossBERT가 만들어질 당시나 평가시 품사나 유의어 정보가 반영되지 않았다. 새로운 정보를 추가했다가 평가시 다시 제외한 것으로 인해 문제가 발생했을 가능성이 있다. 그러나 이것은 하나의 가능성일 뿐, 이 이유만으로 성능이 하락하였다고 보기는

[4]각주 1의 링크

어려울 것이다. 학습과정에서 평가지표들을 확인한 결과 이 역시 GlossBERT에 비해 향상되었다고 보기는 어려웠기 때문이다.

둘째, 유의어 정보 측면에서는 타겟 단어와 유의어 사이에 존재하는 차이가 성능 하락에 영향을 미쳤을 수 있다. 유의어라 하더라도 사용되는 맥락이 상이할 수 있는데, 유의어를 gloss에 포함시켜 정보를 추가하였기 때문이다.

셋째, 품사와 유의어의 정보가, 모델이 문장 context가 아닌 토큰(ambiguous word)정보에 집중하도록 만들었을 가능성을 생각해볼 수 있다. 71.9%라는 하락한 F1 점수는 Token-CLS 방식을 사용한 GlossBERT의 성능과 동일하다. Token-CLS 방식은 타겟 단어의 마지막 hidden state를 사용한 방식으로, 문장 전체를 고려하지 못한 방식이다. 문장 전체를 고려하기 위해서는 타겟 단어의 hidden state가 아닌 [CLS] 토큰의 hidden state를 고려해야하기 때문이다. 그런데 품사와 유의어 정보를 반영한 모델은, 문장이 아닌 토큰만을 고려했을 때의 GlossBERT와 유사한 성능을 보였다. 품사와 유의어 정보를 통해 모델이 타겟 토큰에 집중하게 되었을 가능성이 있다.

세 번째로 제안한 이유는 어떠한 인과관계에 기반된 것은 아니다. 새로운 모델과 Token-CLS 방식의 스코어가 유사하다는 점에서 착안하였다. 그리고 결과 스코어가 비슷하다면 원인도 비슷할 가능성을 이야기해 본 것이다. 그러나 우연의 일치였을 가능성 역시 배제하기 힘들다.

이상, 다의어 구분(WSD)을 위해 품사 정보와 유의어 정보를 추가하여 GlossBERT 모델을 fine-tuning 하였다. F1 score는 71.9%로 기존 GlossBERT 모델에 비해 0.6%p 하락하였다. 그 이유로 이상의 세 가지를 제안하였다.