Abstract

Deep contextualized word representation을 연구(문법이나, 의미론적으로의 단어 사용의 복잡성을 다룸/언어적 맥락을 다룸)

여기서 word vector란 biLM(bidirections language model)의 내부 상태의 학습된 함수 작용으로 볼 수 있음.(large text corpus로부터 학습된 후의 결과)

기존 모델에 결합되어 이 representation 작업이 사용됨. 총 6개의 task에서 성능 향상의 역할을 했다.(QA, TE, SA …등등)/즉 pretrain 시켜서 deep internals를 노출시키는 것이 중요하다.(downstream models과 결합해 semi-supervision 과정.)

1. Introduction

GloVe 같은 word embedding(word representation)은 중요. 하지만 퀄리티를 더 높일 수 있음. 단어의 사용(문법과 의미론)의 복잡성을 다룰 수 있어야 하고 언어적 맥락을 어떻게 다루는지가 중요.

여기서는 새로운 두 문제를 다루는 word representation을 제시.

Entire input sentence를 함수에 넣어 작용해 각 token(단어)가 representation되는 것이 본 논문의 제안이자 다른 점이다.

결과의 vector는 biLSTM으로 오는데, 이 biLSTM은 미리 학습되어진다. 이렇게 미리 학습된 모델을 여기서는 ELMo(Embeddings from Language Models) Representations라 한다.

이전 비슷한 연구의 모델과 달리 deep하다. (all of the internal layers) =>LSTM의 최상단 layer만을 사용하는 것을 넘어 input word의 stacked vector의 선형결합을 배움??이게 뭘까

윗줄의 방식으로 내부 상태를 하는 것은 rich representation 초래. Higher-level LSTM은 맥락 의존적인 word representation을 다루고 특정 task에 쓰일 수 있고, lower-level이라면 part-of-speech tagging에 쓰일 수 있다.

어떤 task에서는 20%의 에러 감소의 성능.

Direct로 비교가 가능한 task??이게 뭐지(수정이 적다는 뜻?)은 비슷한 연구 분야인 CoVe보다 성능이 뛰어나다.

ELMo와 CoVe 둘다 deep representation이 단지 top layer LSTM보다 뛰어난 것을 증명

6개의 task 외에도 좋은 성능을 보일 것이라고 expect

2. related work

Pretrain은 large scale unlabeled text로부터 학습되는데 문법적, 의미론적 의미를 캡쳐한다. 하지만 맥락을 신경쓰지 않은 single representation을 진행해왔다.

최근 연구는 과거 연구를 극복(subword information 추가 or word sense에 대해 separate vector를 학습하여) // 본 모델도 character convolution의 사용으로 subword unit의 이점을 보고 multi-sense information을 downstream task에 통합. 이 부분 헷갈.

Pivot word 주변의 맥락을 encode하는 context-dependent한 context2vec이 있었음. 이러한 pivot word를 representation함과 동시에 supervised 모델 or unsupervised의 encoder로 게산되어 사용하는 경우도 있다. 이 경우 큰 데이터셋을 이용.

본 논문에서도 역시 3천만줄의 문장을 이용.(한 언어의)

또한 layer의 수가 task마다 성능이 다를 수 있다.

Part-of-speech tags(multi task syntactic supervision)의 경우 low-level deep LSTM이면 좋다.

다른 논문에서는 fine-tune하지만 여기서는 하지 않는다. => large,rich, universal biLM representatins가 가능(downstream data size가 작은 경우에도 작동)??이게 뭘까

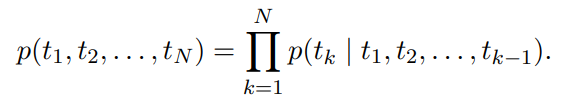
3. ELMo: Embeddings from Language Models

GloVe와 달리 ELMo는 entire input sentence를 기반으로한 함수이다. biLM의 상단의 2개 layer에서 계산되며 character convolution을 이용한다.(즉 내부 network 상태의 linear function이다.)

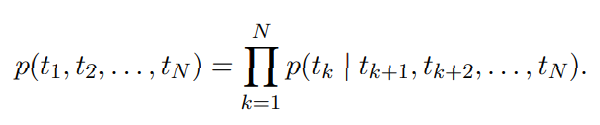
이로써 semi-supervised learning이 가능하다.

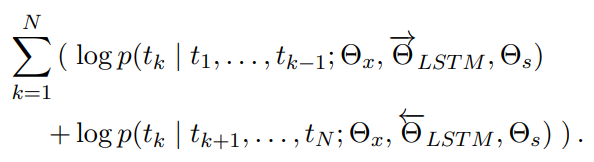
3.1 Bidirectional language models

N개의 token(1 sequence)가 주어지면 그 sequence의 확률을 forward 모델이 계산한다.(token tk의 확률을 계산, t1~tk-1의 history 정보를 이용해서)(본 모델말고 통상적인 수식인가..? 이게 뭘까)



즉 LM 모델에 임베딩된 x가 들어오면 position k에서의 결과는 k+1에서 사용되는데, 즉 최상단 레이어의 결과가 softmax와 함께 k+1때 사용되는 것이다. Bi LSTM이라 역방향 학습시(역전파 말고)에는 반대로

처럼작동 optimization은 다음을 maximization한다.

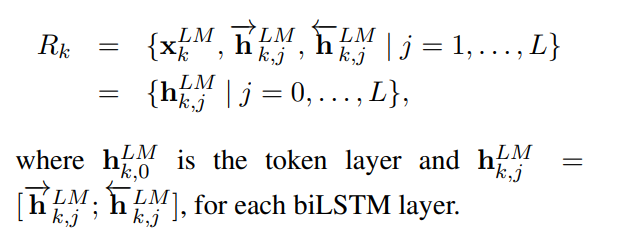


세타x = token representation 세타s의 파라미터 = softmax layer의 파라미터/ 단, 각 방향의 LSTM parameter는 따로 유지(분리해서) 이 식은 Peters 2017의 논문 식과 비슷하지만 독립적으로 파라미터 사용하는 것이 다르다.

아니 그러면 즉, 임베딩(기존 연구로?? X를 만든다..?)을 하고나서 다시 그걸 biLM에 넣고 나온 결과를 다시 downstream에 넣는 것인가?? 이게 뭘까?

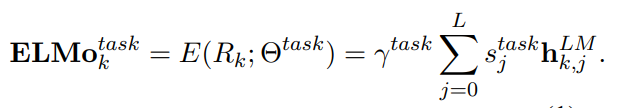
3.2 ELMo

biLM에서 intermediate layer representation의 task specific 결합이 ELMo??



Downstream model에 들어갈 때는 모든 레이어의 Rk들을 collapse한다 하나의 single vector로

단순히 최상단 layer만 선택 = E(Rk) = hk,L이 된다. (TagLM이나 CoVe 처럼) 더 일반적으로 task specific weighting은 다음과 같다. (R= representations, 2L+1개 나옴)

s-task는 softmax-normalized weights고 r-task는 scalar parameter며, 모델이 entire ELMo vector를 scale하도록 한다. R은 optimization 과정을 하도록 돕는다. biLM의 각 activation은 다른 distribution을 가진다 = 어쩔때는 layer normalization을 각 biLM 레이어에 weighting 하기전에 해주어야 한다.

3.3 Using biLMs for supervised NLP tasks