**1. Introduction**

DNN은 input과 target이 고정된 차원의 벡터로 현명하게 인코딩 될 수 있는 문제에만 적용가능했다. Sequence의 길이가 대부분 알려져 있지 않은상태로 잘 표현되기 때문.

논문 아이디어는 하나의 LSTM을 사용하여 한 번에 하나씩 입력 시퀀스를 읽고 큰 고정 차원 벡터 표현을 얻은 다음 다른 LSTM을 사용하여 해당 벡터에서 출력 시퀀스를 추출하는 것입니다.

1. 하나의 LSTM을 사용하여 입력 시퀀스를 한 번에 한 타임스텝으로 읽고 큰 고정 차원 벡터 표현을 얻은 다음 다른 LSTM을 사용하여 해당 벡터에서 출력 시퀀스를 추출하는 것입니다

2. 두 번째 LSTM은 입력 시퀀스에 따라 조건이 지정된다는 점을 제외하고 본질적으로 순환 신경망 언어 모델[28, 23, 30]입니다

장거리 시간 종속성을 가진 데이터를 성공적으로 학습하는 LSTM의 기능은 입력과 해당 출력 사이의 상당한 시간 지연으로 인해 이 응용 프로그램에 자연스럽게 선택됩니다.

LSTM 모델은 입력을 반대로 받는거 주의, 단기 종속성 문제를 쉽게함(source임, target은 그대로 받는듯)

끝내 BLEU 36.5 받음

**2. The model**

RNN은 seq 2 seq 잘했지만, 길이가 다를 경우에는 적용하는 방법이 명확하지 않았음

Simple한 전략으로는 seq to fixed sized vector 를 RNN을 하나 써서 하고

그 뒤에 또 다른 RNN 으로 vector를 sequence로 매핑하는거임

형식적으로 작동은 하지만, 결과적으로 long term dependencies에 빠짐

하지만 LSTM경우에 이러한 문제를 학습가능 하므로, 잘 동작함을 보였음

LSTM의 목표는 조건부 확률을 추정하는 것이다.

LSTM은 먼저 LSTM의 마지막 hidden state에 의해 주어진 고정 차원 표현 v 을 구한 다음 조건부 확률로 계산함 LSTM-LM 공식 사용해서

텍스트이(가) 표시된 사진

자동 생성된 설명

이 식은 모든 단어에 대해 softmax로 표현함.

Graves의 LSTM 공식을 사용함.

각 문장은 <EOS> 토큰으로 끝나야함. 이 기호를 사용함으로, 모든 길이의 시퀀스에 대한 분포 정의 가능 해짐

이 논문에서는 3가지 다른 방법을 사용함

1. 두개의 다른 LSTM 사용, 입력시퀀스용 + 출력 시퀀스용

-> 무시할 수 있는 계산 비용으로 숫자 모델 매개변수가 증가하고, 여러 언어 쌍에 대해 LSTM을 동시훈련 가능하는 것이 자연스러워짐

2. 깊은 LSTM이 얕은 LSTM보다 성능이 우수했음, 그래서 4개의 layer 채택

3. 입력문장의 단어 순서를 바꾸는게 가치있었음. SGD가 입력과 출력 사이에 establish communication 을 쉽게 할 수 있게됨 이를 통해 LSTM 성능 크게 향상

**3. Experiments**

WMT’ 14 English to French dataset 사용

1200백만개 sentences (348M French, 304M English) train 시킴

없는 단어들은 <UNK> 토큰으로 대체

로그 확률 최대화해서 훈련시킴

훈련목표 : 텍스트, 손목시계이(가) 표시된 사진

자동 생성된 설명

source문장 반전될 때 더 잘 학습됨. Target은 그대로

복잡도 5.8 ->4.7

BLEU scores 25.9 -> 30.6

이 현상에 대한 완전한 설명은 없지만 데이터 세트에 대한 많은 단기 종속성이 도입되었기 때문에 발생한다고 생각합니다. 일반적으로 원본 문장을 대상 문장과 연결할 때 원본 문장의 각 단어는 대상 문장의 해당 단어에서 멀리 떨어져 있습니다.

결과적으로 문제는 "minimal time lag"가 큽니다[17]. 원본 문장의 단어를 반대로 하면 원본 언어와 대상 언어의 해당 단어 사이의 평균 거리는 변경되지 않습니다. 그러나 소스 언어의 처음 몇 단어는 이제 대상 언어의 처음 몇 단어에 매우 가깝기 때문에 문제의 최소 시간 지연이 크게 줄어듭니다. 따라서 역전파는 소스 문장과 대상 문장 간의 "통신 설정" 시간을 더 쉽게 가지며 결과적으로 전반적인 성능이 크게 향상됩니다.

Shallow LSTM 보다 deep LSTM이 복잡도 10% 감소

1. 파라미터를 -0.08 ~ 0.08 사이 균일분포로 초기화

2. sgd momentum 사용, lr == 0.7, after 5epochs lr half epoch, 7.5 epoch 동안 학습시킴,

3. 128 batch size 사용

4. 기울기 소실 문제 없었음. 폭발문제는 있었음 -> 임계값 초과할 때 규칙 정해서 방지함

5. 대부분의 문장길이가 짧아서, 미니배치로 128개 랜덤 선택하게 해서, 2배 빠르게 학습하게 함

8개 GPU 사용함

4개는 각각 LSTM, 4개는 softmax 병렬화

128개 미니배치 크기로 초당 6,300단어 속도 달성 -> 총 10일 걸림