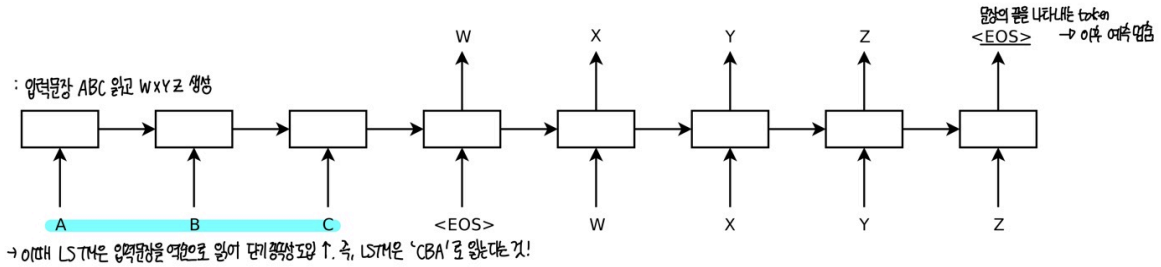
Sequence to Sequence Learning with Neural Networks

Abstract & Introduction

기존 Deep Neural Network는 label 이 지정된 훈련 데이터셋에는 잘 작동하지만, input 과 output 고정된 차원의 벡터로 인코딩 될 수 있는 문제에만 적용될 수 있기 때문에, sequence를 sequence로 mapping 하는 문제 (예를 들어, 기계 번역, 질문 응답과 같은)에는 사용할 수 없음(사전의 길이를 알 수 없기 때문). 본 논문에서는 LSTM(장기 의존성을 학습하는데 강력한 성능을 보이는 RNN 구조)을 이용해서 **입력 sequence를 고정된 차원의 벡터로 매핑한 다음.→ 이때 입력 문장은 전체 sequence, 따라서 단어별이 아닌 문장의 전체 의미를 표현하는 하나의 벡터 aka context vector를 만든다는 것, 이건 sequence 전체의 정보를 포함하는 벡터임** 다른 LSTM을 사용하여 해당 벡터에서 목표 sequence를 디코딩하는 end-to-end sequence 학습 접근법을 제시



하나의 LSTM이 입력시퀀스를 한번에 한 timestep 씩 읽어서 고정된 벡터 표현을 얻고, 또 다른 LSTM을 사용해서 그 벡터에서 출력 시퀀스를 추출

이때 모든 source 문장에서 **단어의 순서를 반대로 변경했을 때 LSTM의 성능이 현저히 향상되었음 → source 문장과 목표 문장 사이에 많은 단기 종속성을 도입한다는 것은, 일반적으로 번역을 할 때 source 문장의 앞쪽 단어는 목표 문장의 앞쪽 단어와 매핑될 가능성이 높음. 하지만 source 문장의 단어의 순서를 바꾸지 않는다면, source 문장의 끝부분의 단어가 목표 문장의 초반부에 번역되는 경우가 있음 (long-distance dependency가 발생), 따라서 번역 과정에서 source 문장을 역순으로 정렬하여 source 문장의 처음부분이 목표 문장의 처음부분과 가깝게 정렬될 확률이 높아지도록 → LSTM이 예측해야 하는 단어간의 timestep 차이가 줄어들어 학습이 쉬워짐**

Related Works

RNN-Language Model, Feedforward Neural Network Language Model은 기계 번역 시스템의 n-best 리스트를 rescoring 하여 기계번역 작업에 주로 품질을 높이는 역할을 했었음

기존의 연구 중, 입력 문장을 CNN을 사용하여 벡터로 매핑한 수 다시 복원하는 방법을 제안했으나 이는 CNN에서 단어 순서 정보가 손실되는 문제가 있었음

또한 기존 입출력을 벡터로 매핑하는 NNLM 모델도 연구되었으나, 직접 번역을 생성하지는 못했었음

Method

Standard RNN은 input과 output의 길이가 다르거나 단조적이지 않을 때 적용하기 힘듦

$$\begin{bmatrix} h_t &= \operatorname{sigm}\left(W^{\mathrm{hx}}x_t + W^{\mathrm{hh}}h_{t-1}\right) \\ y_t &= W^{\mathrm{yh}}h_t \end{bmatrix}$$

입력시퀀스 $(x_1, \cdots, x_T)$ 가주어졌을 때 standard RNN이 출력시퀀스 $(y_1, \cdots, y_T)$ 계산하는 방식

. 이때 x, y의 길이가 같아야 …

따라서 하나의 RNN을 사용하여 입력 시퀀스 → 고정된 벡터로, 그리고 다른 RNN으로 해당 벡터 → 목표 시퀀스로 변환하는 방법이 있으나 이는 long-term dependency 문제로 RNN을 훈련하기 어렵다는 문제가 있음 → **LSTM을 활용** → **RNN의 기울기 소실 문제를 셀 상태와 게이트 구조(망각 게이트)를 이용해서 긴 문장에서도 초반 단어 정보가 잘 유지되도록 함**

LSTM은 장기적인 시간 종속성을 가지는 문제를 해결하는데,

LSTM의 목표 : 주어진 입력 sequence 에서 출력 sequence가 나올 확률을 추정

$$p(y_1, \ldots, y_{T'} \mid x_1, \ldots, x_T) = \prod_{t=1}^{T'} p(y_t \mid v, y_1, \ldots, y_{t-1})$$

출력 sequence / 입력 sequence

$(x_1, \cdots, x_T)$의 마지막 hidden state 이용하여 만든 고정 차원의 vector

이걸 쓰게 hidden state로 설정

각 확률분포는 vocab의 또 단어에 대한 softmax

$P(y_1, \cdots, y_{T'} \mid x_1 \cdots x_T)$: 입력 시퀀스 $(x_1 \sim x_T)$가 주어졌을 때 출력시퀀스 $(y_1 \sim y_{T'})$가 생성될 확률

= 입력 시퀀스 전체 고려한 상태에서, 출력 시퀀스의 각 단어가 순차적으로 등장할 확률을 예측하는 것.

$= P(y_1 \mid v)\, P(y_2 \mid v, y_1)\, P(y_3 \mid v, y_1, y_2) \cdots P(y_{T'} \mid v, y_1, \cdots, y_{T'-1})$

v에서 y_1 나올 확률 / v, y_1에서 y_2 나올 확률 / $v, y_1, \cdots, y_{T'-1}$ 에서 $y_{T'}$ 나올 확률

- LSTM이 입력을 처리할 때, 입력 시퀀스 $(x_1, \cdots, x_T)$ 하나씩 처리하는데 이때 각 입력 단어가 주어질 때마다 이전 상태 h_{t-1} & 현재 입력 x_t → 새 hidden state h_t 갱신
  → 따라서 마지막 은닉상태 h_T는 문장 전체 요약한 벡터 표현 ($h_T = v = h_1'$)
- 디코딩 단계에서, v 를 첫번째 은닉상태로 . $P(y_1 \mid v) = \mathrm{softmax}(W_{y_1} h_1')$ → y_1 예측

조건부 확률 단계별 예측

- → $P(y_2 \mid v, y_1) = \mathrm{softmax}(W_{y_1} h_2')$ → y_2 예측
- → … → $y_{T'}$ 예측

따라서 encoding에서 LSTM, decoding에서 LSTM → 두개의 서로 다른 LSTM을 사용. 이때 매개변수의 개수가 증가하지만, **비용은 거의 들지 않음.** → **두 LSTM이 동시에 계산되는 것이 아니라, 인코더가 먼저 실행된 후 디코더가 실행되기 때문 + 디코더에서 전체 sequence를 한번에 처리하는 것이 아니라 한 단어씩 출력하며 다음 단어를 예측 + context vector는 고정된 크기의 벡터임**

입력 문장의 단어를 **역순으로 변환하는 것이 SGD(확률적 경사하강법)에서 최적화**가 더 잘 됨을 확인 → **SGD는 데이터의 샘플을 랜덤하게 선택하여 미니배치로 학습하고 각 단계에서 손실함수의 기울기를 계산해서 손실함수를 최소화 하는 방향으로 가중치를 업데이트 하여 모델을 학습하는 알고리즘. 이때 SGD는 입력과 출력간 거리가 멀어질 수록 기울기 소실 문제가 발생하기 때문에 단기 의존성이 강한 데이터에 더 잘 작동함.**

Experiments

WMT'14 영어-프랑스어 기계 번역 과제에 적용함. source 는 영어에서 많이 사용되는 16만개 단어, 그리고 목표 는 프랑스어에서 많이 사용되는 8만개의 단어. 이때 어휘에 포함되지 않는 단어 OOV는 UNK 특수 토큰으로 대체

LSTM을 학습 시켜 번역확률(source 문장이 주어졌을 때, 번역 문장 T 의 로그 확률)을 최대화 하는 방식으로 훈련.

이때 Beam Search Decoding이란, **Seq2Seq 모델에서 가장 가능성이 높은 출력 문장을 찾기 위해 사용되는 탐색 알고리즘으로 매 timestep 마다 B개의 가장 가능성 높은 문장 조합을 유지함**. → **번역이 진행 중인 문장 B개를 유지하고 매 timestep에서 현재 beam에 있는 각 문장을 새 어휘 단어와 결합하여 확장하는데 이때 급격히 증가한 가능한 후보의 개수 중 모델의 log probability가 높은 상위의 B개의 가설만 유지됨.** 이때 beam size가 1일 때 (beam size가 작다는 건, 탐색 공간이 작다는 것)도 좋은 성능을 보임

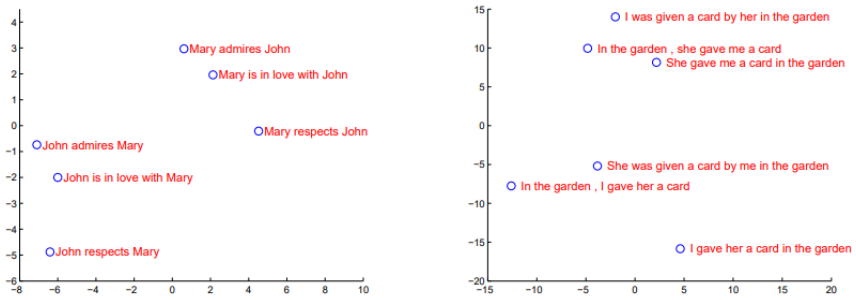
기존 SMT(통계적 기계 번역) 점수 + LSTM 점수의 평균값으로 새로운 점수를 계산 → 기존 SMT 보다 더 나은 번역을 선택할 수 있도록

Deep LSTM (4 layers, 1000 cells each layer) 사용 → 이때 가장 결과가 좋은 모델은 random initialization과 minibatch random order 가 서로 다른 LSTM 을 앙상블 했을 때

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
Best WMT'14 result [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	~45

LSTM은 OOV 단어를 처리하지 못한다는 단점에도, SMT보다 높은 성능을 보임



LSTM이 문장 처리한 수 hidden state를 PCA로 시각화 한 것

이때 각 문장의 의미에 따라 cluster가 형성된 것이 확인됨(좌 우 그래프 속 문장들의 의미가 동일) → 단어의 순서 중요, 그리고 능동과 수동의 변화에는 둔감함이 확인됨 (I gave her와 I was given by her이 같은 cluster)

Discussion & Conclusion

단순한 LSTM 기반 접근 방식이 기계번역에서 성공적이었다는 것은 훈련 데이터가 충분하다면 다른 시퀀스 학습 문제에도 효과적일 가능성 시사

source 문장을 역순으로 취하는 것만으로도, 학습 성능이 크게 향상됨

LSTM이 상대적으로 최적화 되지 않은 상태에서도 SMT보다 높은 성능 → 추가적인 연구가 필요함