Sequence to Sequence Learning with Neural Networks

Ilya Sutskever & Oriol Vinyals & Quoc V. Le (NIPS 2014)

딥러닝 논문리뷰
김영민 교수님
한양대학교 산업 데이터 엔지니어링학과
석사과정 강병모

한양대학교

2022-10-04

Introduction- Statistical Machine Translation(SMT)

Language Model

- 언어 모델은 문장(Sequence)에 확률을 부여하는 모델

- 언어 모델을 통해 특정 상황에서 적절한 문장, 단어를 확률로써 예측

  ex) P(나는 점심을 먹었다|I had lunch)>P(나는 화가 난다|I had lunch)

  P(먹었다|나는 점심을)>P(화가 난다|나는 점심을)

- 하나의 문장(W)은 여러 개의 단어(w)로 구성

  -> $P(W) = P(w_1, w_2, \ldots, w_T)$

  -> P(나는 점심을 먹었다)= P(나는, 점심을, 먹었다)

- 연쇄 법칙(Chain Rule)

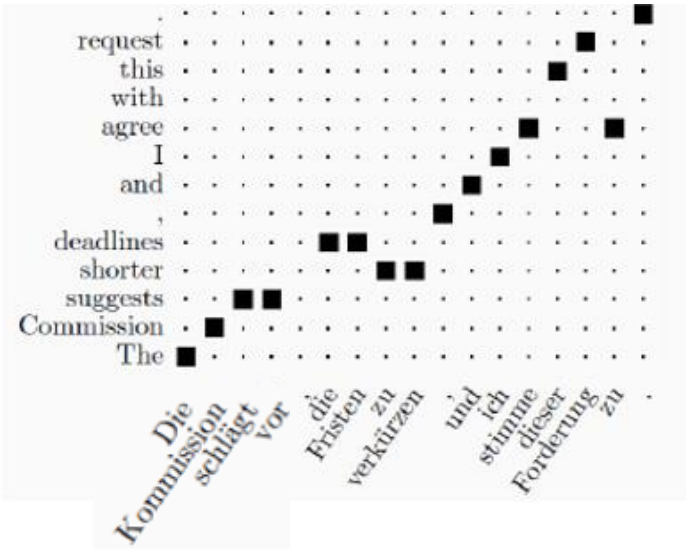
$$P(w_1, w_2, \ldots, w_T) = P(w_1) * P(w_2|w_1) * P(w_3|w_1, w_2) * \cdots P(w_T|w_1, w_2 \ \ldots \ w_{T-1})$$

-> $\prod_{i=1}^{n} P(w_i \mid w_1, w_2, \ldots, w_{i-1})$

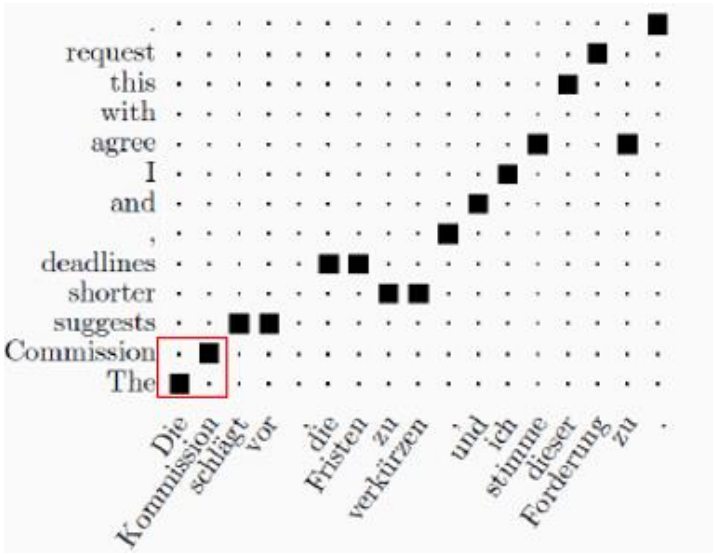
-> P(나는 점심을 먹었다)= P(나는)* P(점심을|나는)* P(먹었다|나는 점심을)

Introduction- Statistical Machine Translation(SMT)

Word Base SMT



Phrase Base SMT



Translation Model

- 번역모델은 번역할 언어(Source language)와 번역될 언어(Target language)로 구성
- 번역모델은 병렬 말뭉치(Corpus)가 필요
- 번역모델을 통해 alignment를 추출

Introduction- Statistical Machine Translation(SMT)

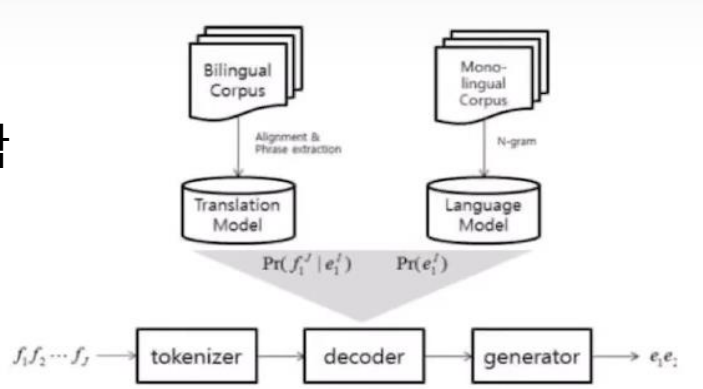
Statistical Machine Translation

- 대용량 corpus에서 학습된 통계 정보를 활용함
- 언어모델과 번역모델을 나누어서 번역 수행
- 한계점

1. Local context(Only phrase)

   -구를 넘어선 문장 관계를 표현할 수 없음

2. Language 모델과 translation 모델 각각 최적화 및 모두 학습
3. Model size : Big
4. CPU



$$\text{argmax}_y P(x|y)P(y)$$

x : source sentence

y : target sentence

$P(x|y)$: translation model

$P(y)$: language model

Introduction- Neural Machine Translation(NMT)

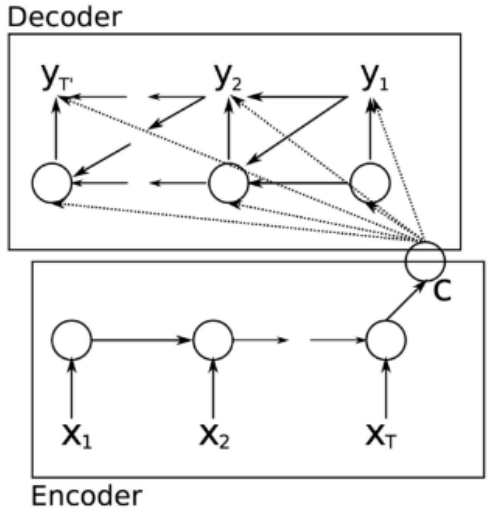
Neural Machine Translation

- Single neural network를 통해 source 문장에서 target 문장의 다음 단어가 나타날 확률을 예측
- Encoder & decoder의 구조
- SMT와 비교 했을 때 장점

1. Global context(Whole sentence)

   -문장 전체 표현가능

2. Global optimization
3. Only parallel corpus
4. Model size : Relatively small
5. GPU



$$P(y|x) = P(y_1|x)P(y_2|y_1,x) \\ P(y_3|y_1,y_2,x) \dots \\ P(y_T|y_1,\dots,y_{T-1},x)$$

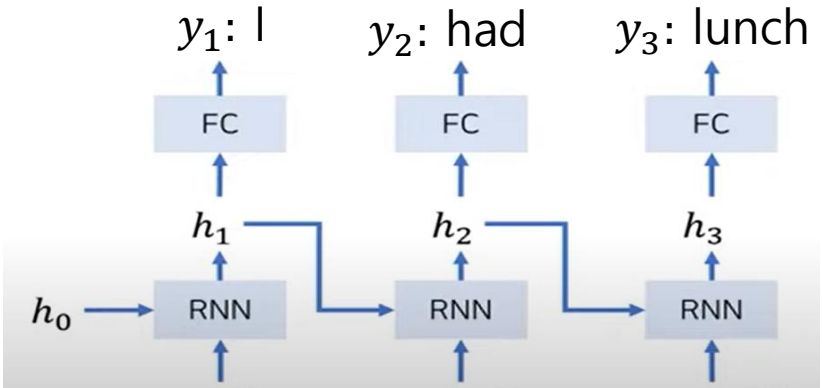
x : source sentence

y : target sentence

$P(y|x)$: 찾고자 하는 번역 모델

Introduction- Traditional RNN Based Machine Translation

- 초기 RNN 기반 기계 번역
  - 입력과 출력 크기 같다고 가정
    - Input :$(x_1, \dots, x_t)$
    - Output: $(y_1, \dots, y_t)$

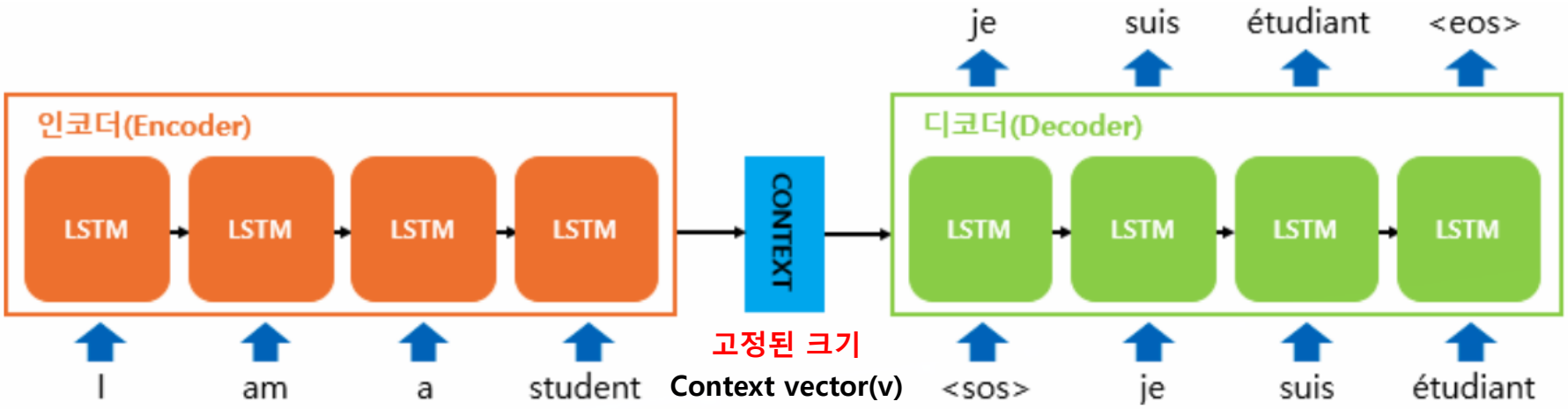


$$h_t = \text{sigm}\left(W^{\text{hx}} x_t + W^{\text{hh}} h_{t-1}\right)$$
$$y_t = W^{\text{yh}} h_t$$

- 한계점
  - Input size와 output size가 다르다면, 좋은 성능을 보일 수 없음
  - 한국어와 영어와 같이 어순이 다르다면, 좋은 성능을 보일 수 없음

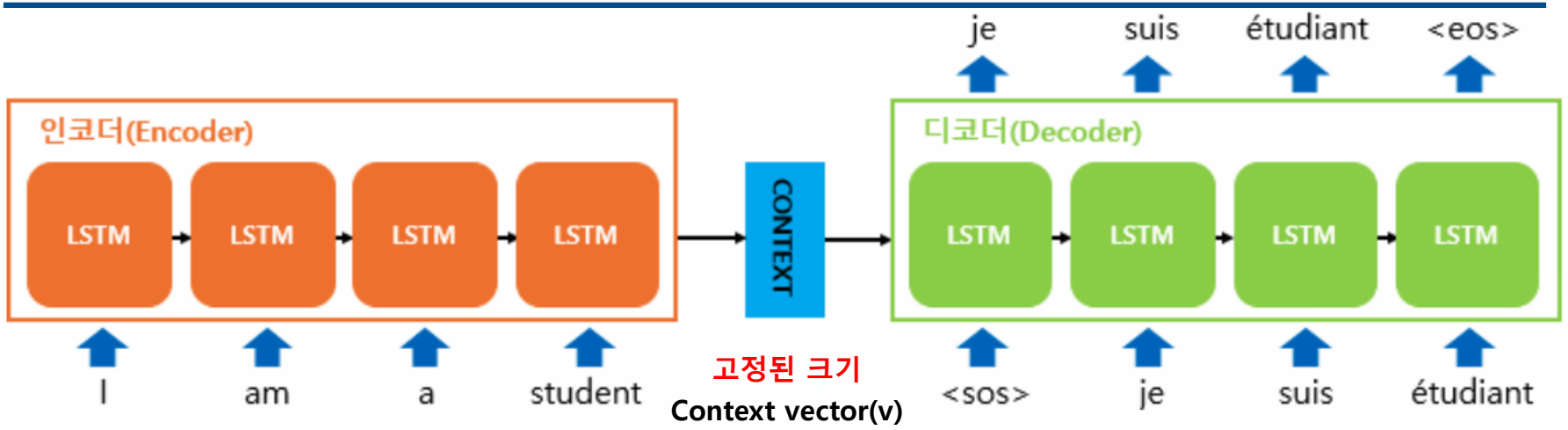
- 논문에서 해결책 : LSTM(Long Short Term Memory), context vector, reversing the input sequence

Main Model - Sequence to Sequence



- Multi-layer LSTM (논문에서 4 Layer LSTM)
- Context Vector
- Reversing the input sequence
  - SMT BLEU score : 33.3
  - BLEU score : 34.8

Main Model - Sequence to Sequence



Whole source input sentence

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

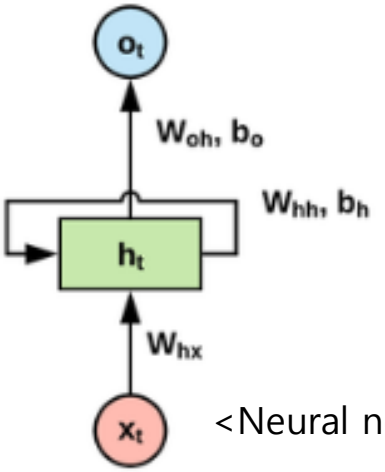
Target sentence(French) Source sentence(English)

-> p(je, suis, étudiant| I, am a student)

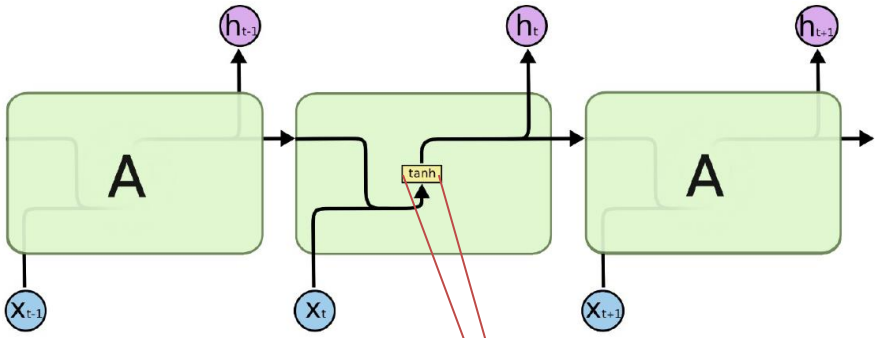
= p(je|v, <SOS>)* p(suis|v, <sos>, je)* p(étudiant|v,<sos>, je, suis>

Main Model - Sequence to Sequence

Why LSTM?

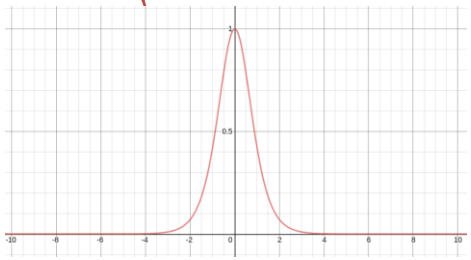


<Neural network>



<RNN>

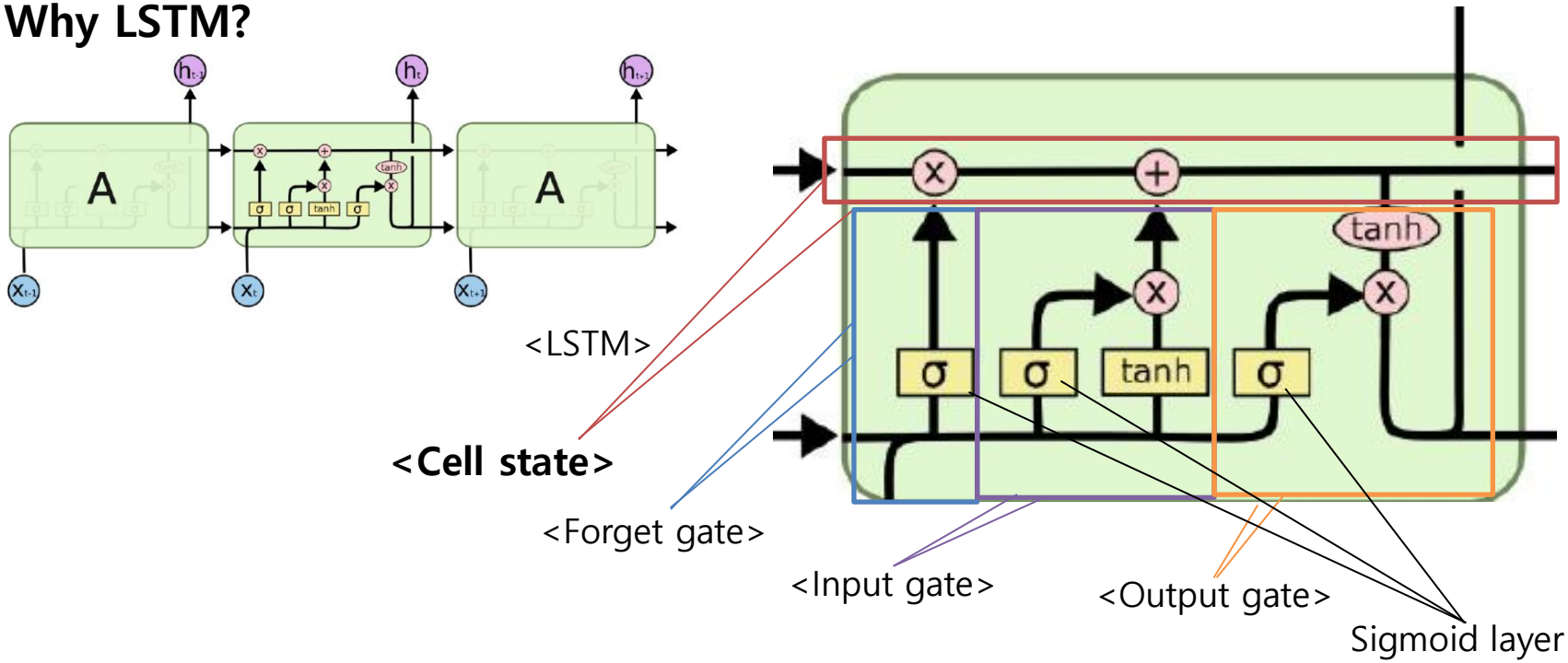
- RNN
  - ✓ 신경망(Neural network) 모듈이 반복되는 형태
    => 출력 벡터가 다시 입력됨
- RNN 장점
  - ✓ 어떠한 sequential 한 데이터도 처리할 수 있음
- RNN의 단점
  - ✓ Long term dependency
    -> 문장(Sequence)이 길어지면 vanishing gradient 문제
    => Sequence 앞쪽 hidden state 정보 반영하지 못함



<tanh 미분함수>
-> 길이가 길어지면, 멀리 있는 hidden state정보가 소실됨

Main Model - Sequence to Sequence

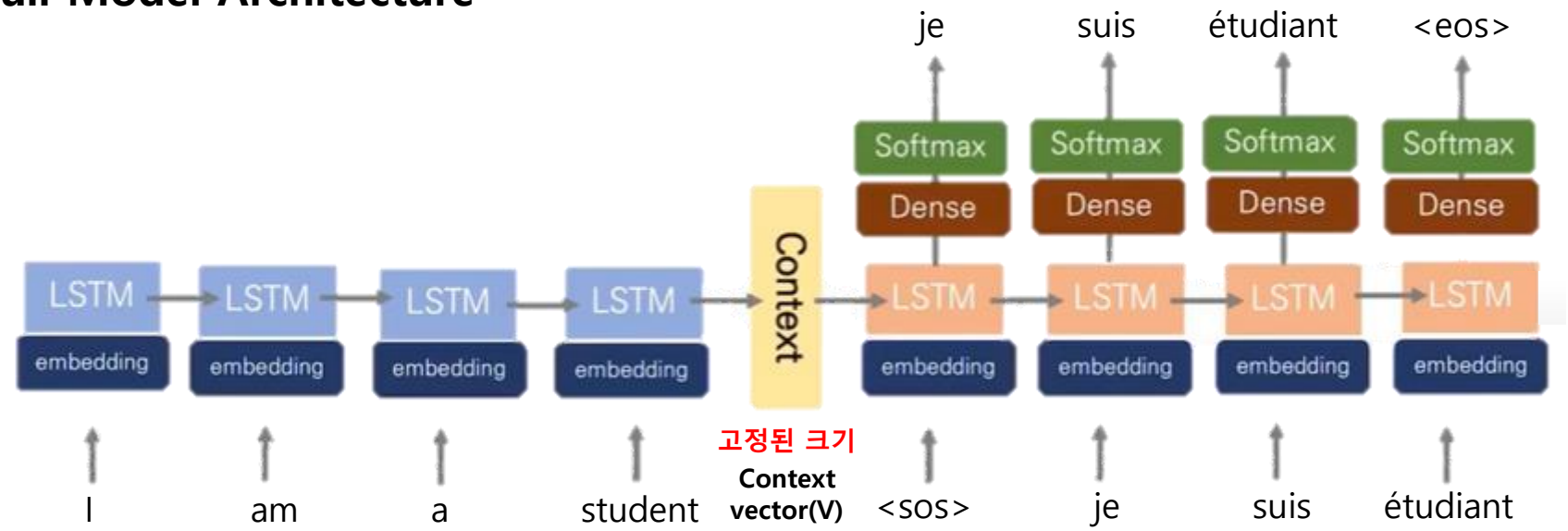
Why LSTM?



- LSTM을 통해 개선
  - ✓ RNN hidden state에 cell state를 추가하여 information을 추가하거나 삭제
  - ✓ Cell state와 hidden state의 재귀적으로 계산을 통해 long-term dependency 해결

Main Model – Model Architecture

Full Model Architecture

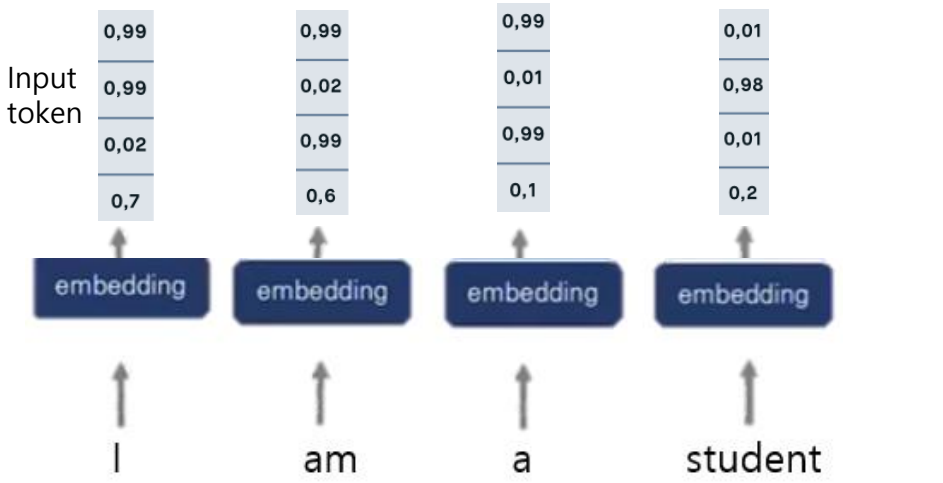


➢ 모델 주요 구성 요소

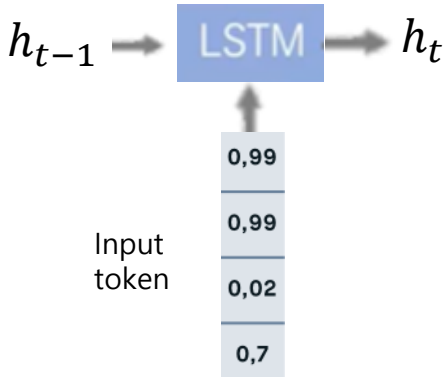
1. Embedding
2. Encoder
3. Decoder
4. Dense & Softmax Layer

Main Model – Model Architecture

Embedding & Encoder



<Embedding>



<Encoder>

- Embedding
  - ✓ 컴퓨터가 이해할 수 있는 구조로 변환

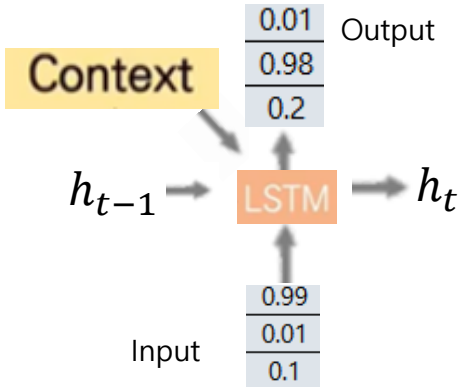
    -> Embedding 된 값->벡터로 표현

- Encoder
  - ✓ Embedding된 단어와 hidden state는 LSTM cell을 통해 연산

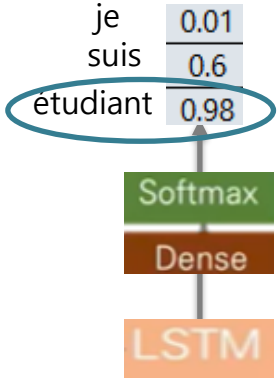
    ->Hidden state update
  - ✓ 마지막 hidden state->context vector

Main Model – Model Architecture

Decoder, Dense & Softmax Layer



<Decoder>



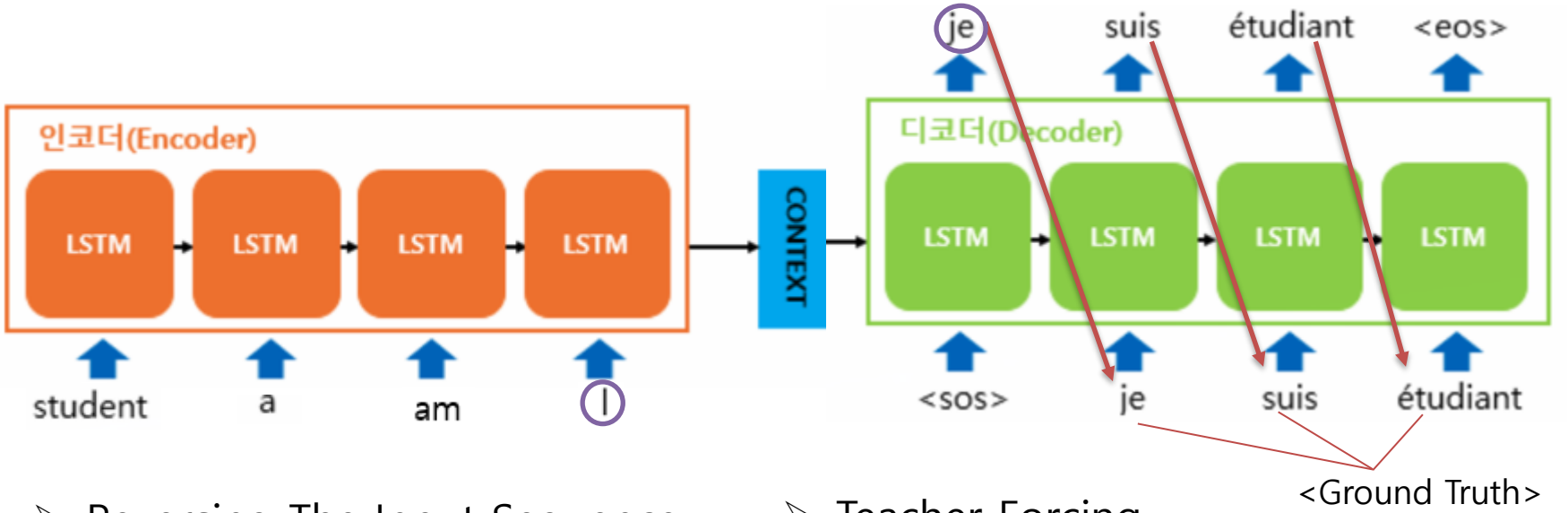
<Softmax & Dense >

- Decoder
  - ✓ Embedding 단어와 hidden state, context vector는 LSTM cell를 통해 연산
    - -> Hidden state update
    - -> 다음에 나올 단어 확률 예측

- Dense & Softmax layer
  - ✓ Dense, softmax layer를 거쳐 확률이 제일 높은 단어 선택
    - => 확률 제일 높은 étudiant 선택

Model Training

Reversing The Input Sequence & Teacher Forcing

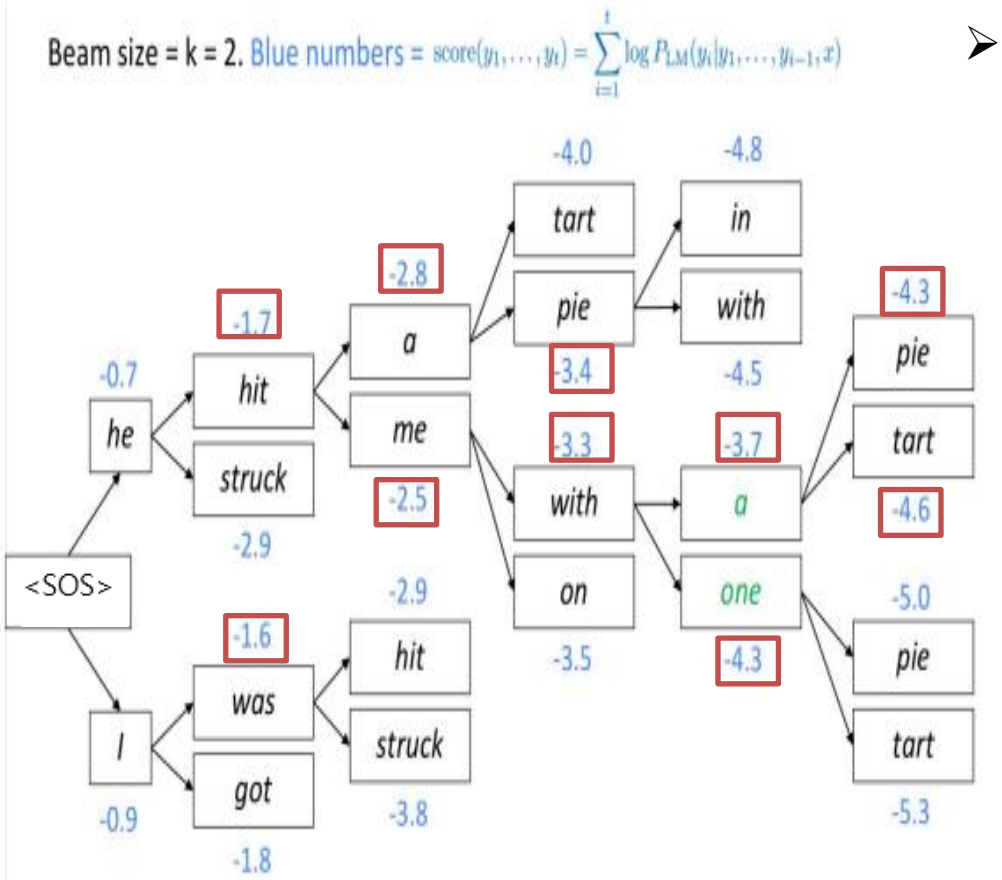


- Reversing The Input Sequence
  - ✓ Source sentence의 순서를 거꾸로 사용하여 번역 성능 향상
    -> BLEU score 25.9->30.6
  - ✓ Why?
    -> Source sentence의 첫 단어와 target sentence의 첫 단어와의 거리가 가까워지기 때문

- Teacher Forcing
  - ✓ 실제 정답(Ground truth)를 decoder의 다음 input으로 넣어주는 기법
  - ✓ Why?
    -> Decoder의 잘못된 예측으로 인해, 학습속도가 저하되는 것을 막기 위해
  - ✓ 실제 정답 비율 높아지면, 학습 속도 up
    -> 하지만, train data overfitting 확률 up

Inference- Beam Search

Beam Search



- Beam Search
  - ✓ 각 단계에서 k개의 가장 가능도가 높은 토큰들로 유지하며 다음 단계를 탐색하는 방법
  - ✓ K : 사용자 지정 hyper parameter
    - ->K가 커지면 더 좋은 target sequence 생성
    - ->하지만, 속도 느려짐
  - ✓ <SOS>부터 시작해서 <EOS>를 만날 때까지 계속 진행됨
  - ✓ 최적의 target sentence를 보장하는 것은 아님

Model Evaluation-BLEU Score

BLEU Score(Bilingual Evaluation Understudy)

$$BLEU = min(1, \frac{output\ length(\text{예측 문장})}{reference\ length(\text{정답 문장})})(\prod_{i=1}^{4} precision_i)^{\frac{1}{4}}$$

짧은 문장이 더 좋은 점수를 받는 것을 방지하고자 예측 문장 단어 수를 실제 문장 단어 개수로 나눔 (brevity)

같은 단어 여러 번 반복할 때 보정(clipping) 후, n-gram(1~4)을 통해 순서쌍이 얼마나 겹치는지 계산(precision)

- Source sentence를 받아 디코딩한 문장과 실제 정답 번역문장이 얼마나 유사한지 비교하여 번역에 대한 성능을 측정하는 지표 (점수가 높을수록 좋음)

- 점수 계산 요소
  - ✓ Precision : 얼마나 겹치는지 (n-gram, n=1~4)
  - ✓ Clipping : 같은 단어 연속적으로 반복해서 나올 때, 점수 보정
  - ✓ Brevity penalty : 짧은 문장에 대한 점수 보정

정답문장 : 나는 열정적인 학생이다.
예측문장 : 나는 나는 나는 열정적인 대학원생이다.

1-gram precision(clipping x)

$\rightarrow \frac{\text{일치하는 } 1\text{-}gram \text{ 수}}{\text{모든} 1\text{-}gram \text{ 수}} = \frac{4}{5}$

1-gram precision (clipping 0)

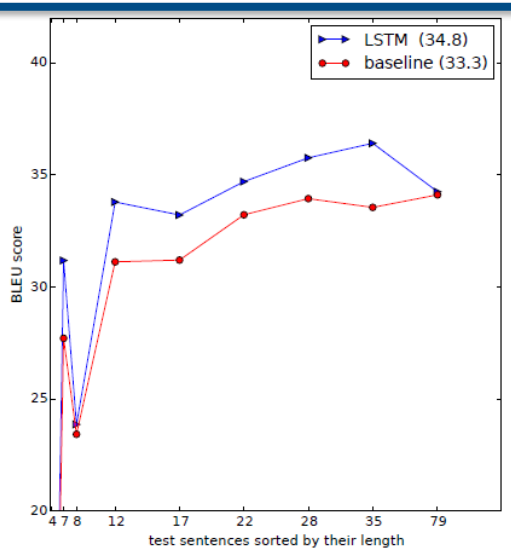
$\rightarrow \frac{\text{일치하는 } 1\text{-}gram \text{ 수}}{\text{모든} 1\text{-}gram \text{ 수}} = \frac{2}{5}$

=> 문장의 길이와 단어의 중복을 고려한 정답문장과 예측문장 사이의 겹치는 정도를 계산하는 지표

Experimental Results

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**

<Figure 1>



<Figure 2>

- Task : WMT' 14 English to French
- Figure 1
  - ✓ Input sequence 문장을 거꾸로 넣었을 때 성능 좋음
  - ✓ Beam size가 커지면 성능 좋음
- Figure 2
  - ✓ 문장이 길어져도 BLEU score 잘 나옴
  - ✓ 하지만, 한 문장에 35단어가 넘어가면 급격하게 성능 감소

Experimental Results

<Figure 3>

- Figure 3
  - ✓ Encoder로 embedding 한 결과를 PCA를 통해 차원 축소한 결과
    - ✓ 유사한 문장(단어의 순서, 주어/동사의 의미)에 따라 민감
    - ✓ 수동, 능동 형태는 큰 영향을 받지 않음

=> 문장의 의미를 반영한 context vector가 잘 형성됨

Conclusion

- RNN 기반 encoder-decoder 구조 대신 LSTM 사용
  - ✓ Long term dependency(vanishing gradient) 문제 개선
- Context vector 사용
  - ✓ Input size와 output size가 같을 때 발생했던 문제 개선
- Reversed order input sequence를 사용
  - ✓ 번역 성능 향상
- 한계점
  - ✓ 단어의 개수가 35개 넘어가면 성능 약화

    => Attention 등장