Speech Style CopyCAT

Seq2Seq with Attention

강태형 T4IR 2019/09/20

목차

NAVER

00

Idea Select:
why start this project?

NAVER

WHY START THIS PROJECT?

1. 배웠던 LSTM 모델을 실제로 적용해보고 싶음
2. 번역기나 챗봇은 너무 식상해서 재미가 없을 것 같음
3. Vanila Seq2Seq Model보다 Challenging 한 모델을 다뤄보고 싶음

‘말투’를 따라하는 모델을 만드는 건 어떨까?

01

Data & Model description

NAVER

Data description : 성경 자료

성경에는 여러가지 번역이 존재

고어로 쓰여 있는 1961년에 번역본부터

현대어로 쓰여있는 다양한 버전이 있음

창1:1 태초에 하나님이 천지를 창조하시니라
창1:2 그 땅이 혼돈하고 공허하며 흑암이 깊음 위에 있고 하나님의 영은 수면 위에 운행하시니라
창1:3 하나님이 이르시되 빛이 있으라 하시니 빛이 있었고
창1:4 그 빛이 하나님이 보시기에 좋았더라 하나님이 빛과 어둠을 나누사
창1:5 하나님이 빛을 낮이라 부르시고 어둠을 밤이라 부르시니라 저녁이 되고 아침이 되니 이는 첫째 날이니라
창1:6 하나님이 이르시되 물 가운데에 궁창이 있어 물과 물로 나뉘라 하시고
창1:7 하나님이 궁창을 만드사 궁창 아래의 물과 궁창 위의 물로 나뉘게 하시니 그대로 되니라
창1:8 하나님이 궁창을 하늘이라 부르시니라 저녁이 되고 아침이 되니 이는 둘째 날이니라
창1:9 하나님이 이르시되 천하의 물이 한 곳으로 모이고 뭍이 드러나라 하시니 그대로 되니라
창1:10 하나님이 뭍을 땅이라 부르시고 모인 물을 바다라 부르시니 하나님이 보시기에 좋았더라
창1:11 하나님이 이르시되 땅은 풀과 씨 맺는 채소와 각기 종류대로 씨 가진 열매 맺는 나무를 내라 하시니 그대로 되어
창1:12 땅이 풀과 각기 종류대로 씨 맺는 채소와 각기 종류대로 씨 가진 열매 맺는 나무를 내니 하나님이 보시기에 좋았더라

파일(F) 편집(E) 서식(O) 보기(V) 도움말
창1:1 태초에 하나님이 우주를 창조하셨다.
창1:2 지구는 아무 형태도 없이 텅 비어 흑암에 싸인 채 물로 뒤덮여 있었고 하나님의 영은 수면이 활동하고 계셨다.
창1:3 그때 하나님이 '빛이 있으라' 라고 말씀하시자 빛이 나타났다.
창1:4 그 빛은 하나님이 보시기에 좋았다. 하나님이 빛과 어두움을 나누어
창1:5 빛을 낮이라 부르시고 어두움을 밤이라고 부르셨다. 저녁이 지나고 아침이 되자 이것이 첫째 날이었다.
창1:6 하나님이 '물 가운데 넓은 공간이 생겨 물과 물이 나누어져라' 하시자 그대로 되었다. 이렇게 하나님은 공간을 만들
창1:7 (6절과 같음)
창1:8 그 공간을 하늘이라고 부르셨다. 저녁이 지나고 아침이 되자 이것이 둘째 날이었다.
창1:9 하나님이 '하늘 아래 있는 물은 한 곳으로 모이고 뭍이 드러나라' 하시자 그대로 되었다.
창1:10 하나님은 뭍을 땅이라 부르시고 모인 물을 바다라고 부르셨다. 이것은 하나님이 보시기에 좋았다.
창1:11 하나님이 '땅은 온갖 채소와 씨 맺는 식물과 열매 맺는 과일 나무들을 그 종류대로 내어라' 하시자 그대로 되었다
창1:12 이렇게 땅이 온갖 채소와 씨 맺는 식물과 열매 맺는 과일 나무들을 그 종류대로 내니 하나님이 보시기에 좋았다.

Data description : 성경 자료

각 다른 버전의 성경을

번역기 모델의 source / target data로

다루면 말투를 따라할 것이라 생각!

창1:1 태초에 하나님이 천지를 창조하시니라
창1:2 그 땅이 혼돈하고 공허하며 흑암이 깊음 위에 있고 하나님의 영은 수면 위에 운행하시니라
창1:3 하나님이 이르시되 빛이 있으라 하시니 빛이 있었고
창1:4 그 빛이 하나님이 보시기에 좋았더라 하나님이 빛과 어둠을 나누사
창1:5 하나님이 빛을 낮이라 부르시고 어둠을 밤이라 부르시니라 저녁이 되고 아침이 되니 이는 첫째 날이니라
창1:6 하나님이 이르시되 물 가운데에 궁창이 있어 물과 물로 나뉘라 하시고
창1:7 하나님이 궁창을 만드사 궁창 아래의 물과 궁창 위의 물로 나뉘게 하시니 그대로 되니라
창1:8 하나님이 궁창을 하늘이라 부르시니라 저녁이 되고 아침이 되니 이는 둘째 날이니라
창1:9 하나님이 이르시되 천하의 물이 한 곳으로 모이고 뭍이 드러나라 하시니 그대로 되니라
창1:10 하나님이 뭍을 땅이라 부르시고 모인 물을 바다라 부르시니 하나님이 보시기에 좋았더라
창1:11 하나님이 이르시되 땅은 풀과 씨 맺는 채소와 각기 종류대로 씨 가진 열매 맺는 나무를 내라 하시니 그대로 되어
창1:12 땅이 풀과 각기 종류대로 씨 맺는 채소와 각기 종류대로 씨 가진 열매 맺는 나무를 내니 하나님이 보시기에 좋았더라

파일(F) 편집(E) 서식(O) 보기(V) 도움말
창1:1 태초에 하나님이 우주를 창조하셨다.
창1:2 지구는 아무 형태도 없이 텅 비어 흑암에 싸인 채 물로 뒤덮여 있었고 하나님의 영은 수면이 활동하고 계셨다.
창1:3 그때 하나님이 '빛이 있으라' 라고 말씀하시자 빛이 나타났다.
창1:4 그 빛은 하나님이 보시기에 좋았다. 하나님이 빛과 어두움을 나누어
창1:5 빛을 낮이라 부르시고 어두움을 밤이라고 부르셨다. 저녁이 지나고 아침이 되자 이것이 첫째 날이었다.
창1:6 하나님이 '물 가운데 넓은 공간이 생겨 물과 물이 나누어져라' 하시자 그대로 되었다. 이렇게 하나님은 공간을 만들
창1:7 (6절과 같음)
창1:8 그 공간을 하늘이라고 부르셨다. 저녁이 지나고 아침이 되자 이것이 둘째 날이었다.
창1:9 하나님이 '하늘 아래 있는 물은 한 곳으로 모이고 뭍이 드러나라' 하시자 그대로 되었다.
창1:10 하나님은 뭍을 땅이라 부르시고 모인 물을 바다라고 부르셨다. 이것은 하나님이 보시기에 좋았다.
창1:11 하나님이 '땅은 온갖 채소와 씨 맺는 식물과 열매 맺는 과일 나무들을 그 종류대로 내어라' 하시자 그대로 되었다
창1:12 이렇게 땅이 온갖 채소와 씨 맺는 식물과 열매 맺는 과일 나무들을 그 종류대로 내니 하나님이 보시기에 좋았다.

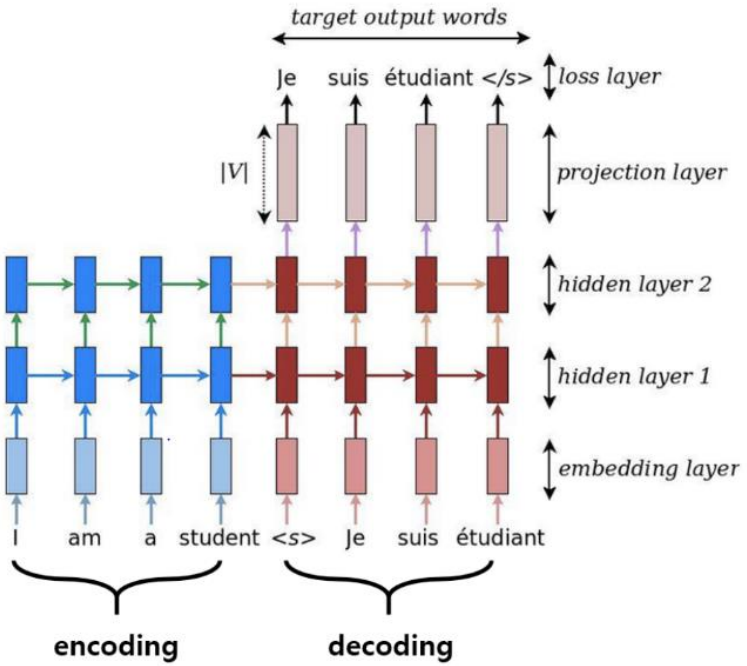
Machine Translation model의 기본: Seq2Seq

1. Seq2Seq2 Model

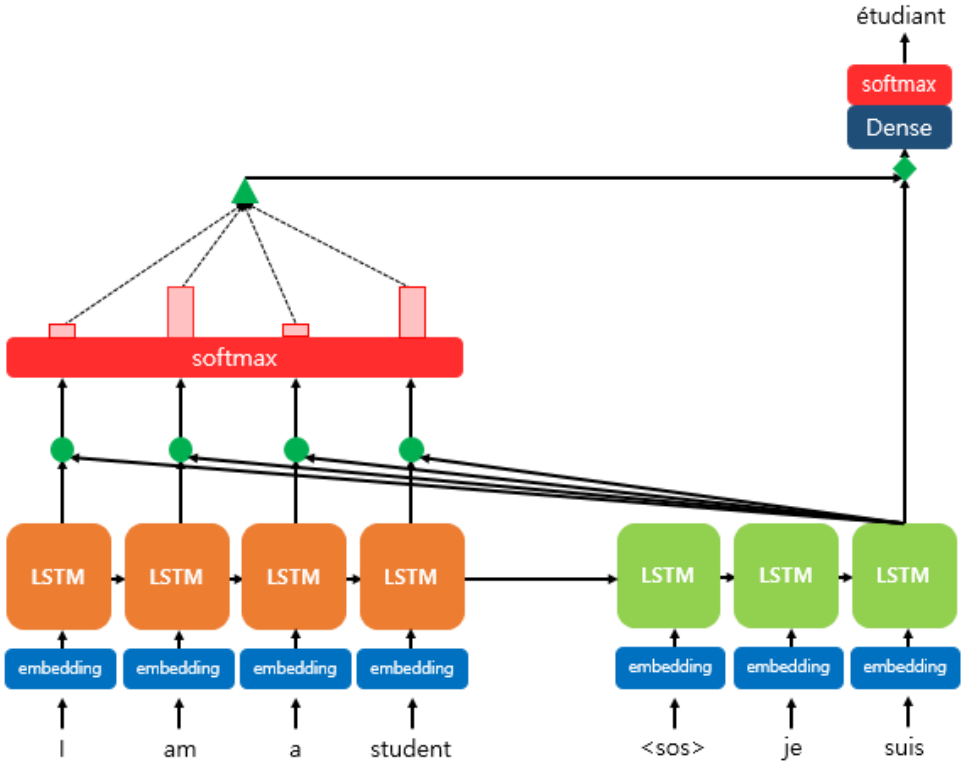
Encoder - Decoder 형식의 두 LSTM 모델 사용

Encoder에서 생성된 Context vector

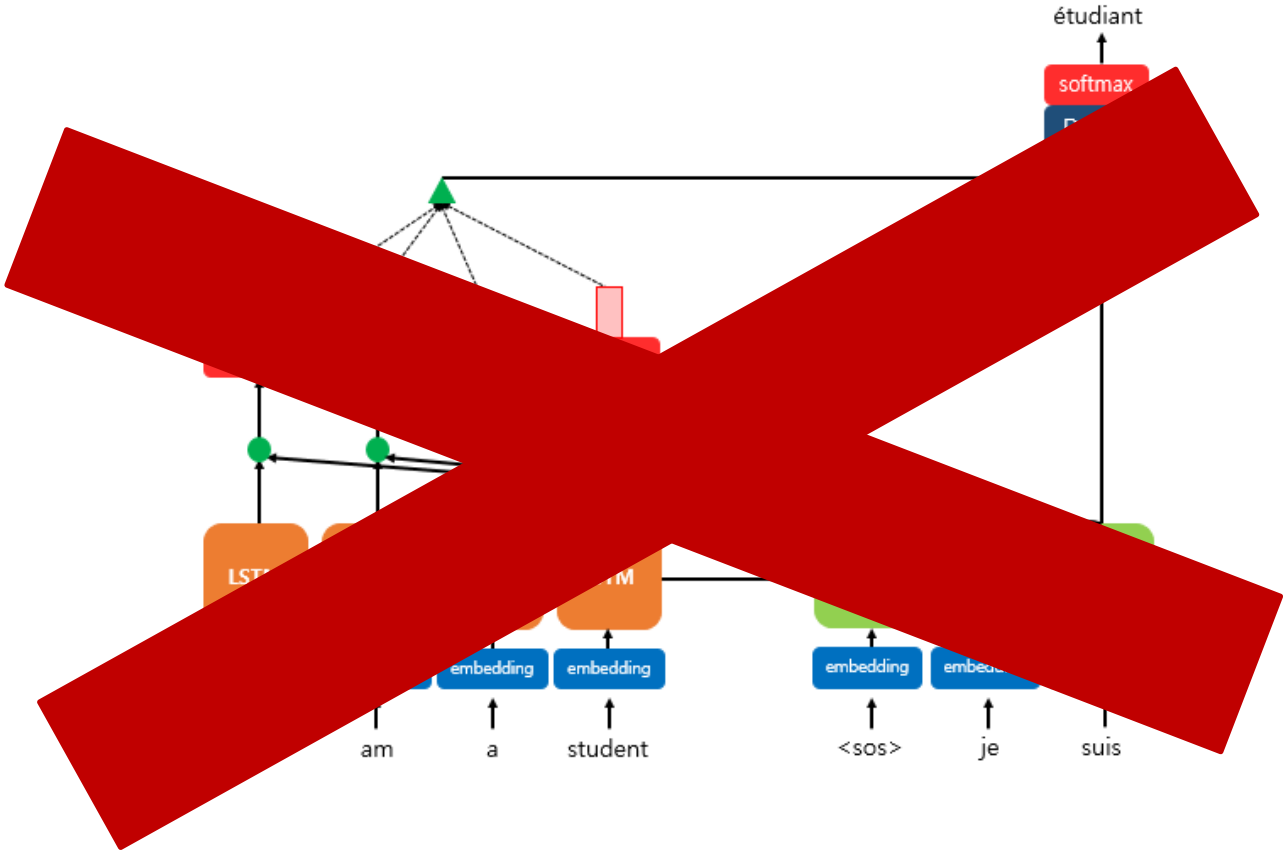
⇒ Decoder로 전달되어 학습 및 예측 수행



+ Attention


étudiant
softmax
Dense
softmax
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
embedding
embedding
embedding
embedding
embedding
embedding
embedding
I
am
a
student
<sos>
je
suis

+ Attention


étudiant
softmax
embedding
embedding
embedding
am
a
student
<sos>
je
suis

+ Attention

전체 입력 문장을 전부 다 동일한 비율로 참고하는 것이 아니라,
해당 시점에서 예측해야할 단어와 연관이 있는
입력 단어 부분을 좀 더 집중(attention)해서 보는 것!

⇒ Context Vector를 attention을 적용해 매 time step마다 갱신

02

Training

NAVER

Training

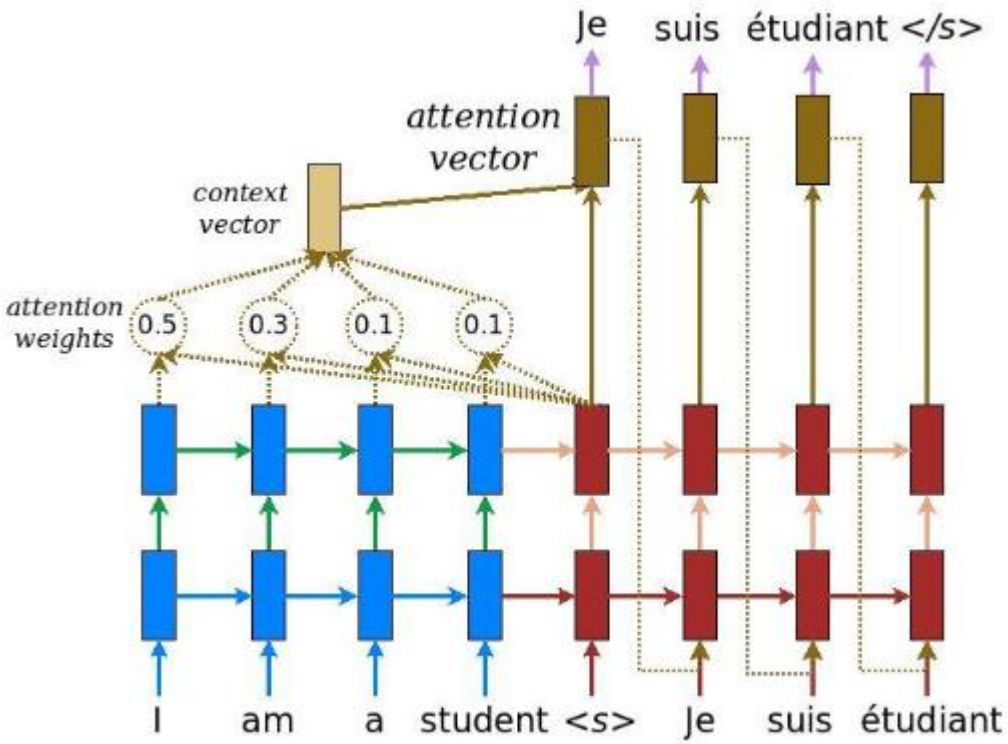
Prototype으로 keras의 tokenizer class를
사용해 단순히 띄어쓰기만으로
단어를 분류,Vocabulary를 만들었음

```
[ ]  print ("Input Language; index to word mapping")
     convert(inp_lang, input_tensor_train[0])
     print ()
     print ("Target Language; index to word mapping")
     convert(targ_lang, target_tensor_train[0])
```

```
Input Language; index to word mapping
2 ----> <start>
44 ----> 그때
7532 ----> 블레셋군은
19690 ----> 아벡에
31554 ----> 집결하였고
1663 ----> 이스라엘군은
3315 ----> 이스르엘
3719 ----> 계곡
10791 ----> 샘
521 ----> 곁에
973 ----> 진을
1570 ----> 쳤다
1 ----> .
3 ----> <end>

Target Language; index to word mapping
1 ----> <start>
204 ----> 블레셋
518 ----> 사람들은
22 ----> 그들의
7 ----> 모든
815 ----> 군대를
19831 ----> 아벡에
32912 ----> 모았고
30 ----> 이스라엘
518 ----> 사람들은
8282 ----> 이스르엘에
27 ----> 있는
4927 ----> 샘
444 ----> 곁에
1164 ----> 진
5448 ----> 쳤더라
2 ----> <end>
```

Training



좌측의 모델과 같은

컨셉으로 학습

03

Conclusion

NAVER

Conclusion

```
<tensorflow.python.training.tracking.util.CheckpointLoadStatus at 0x7f40aaa699e8

[ ]  translate(u'야 이런')

input: <start> 야 이런 <end>
Predicted translation: 그는 내게 대한 일을 가르치리니 <end>
```

⇒ 결론 : 단순 Tokenizing 한 모델로 학습은 역부족

⇒ 번역이 아니라 아예 새로운 말을 하고 있음

To be better…

⇒ Data의 어근 추출

⇒ 단어 의미 전달을 더 명확하게 하자

⇒ Konlpy.tag.Okt

```
[9]  from konlpy.tag import Okt

     okt = Okt()

     # make a space between words and puncts for input
     def preprocess_sentence_input(w):
       w = re.sub(r'([?.!,])', r' #1 ', w)
       w = re.sub(r'[" "]+', " ", w)

       # replacing everything and with space except (가-힣, ".", "?", "!", ",")
       w = re.sub(r"[^가-힣?.!,]+", " ", w)

       w = w.rstrip().strip()

       w = ' '.join(okt.morphs(w, stem=True))

       # adding a start and an end token to the sentence
       # so that the model know when to start and stop predicting.
       w = '<start> ' + w + ' <end>'
       return w
```

To be better…

⇒Okt.morph(stem=True)

```
Input Language; index to word mapping
4 ----> <start>
8 ----> 의
239 ----> 로운
252 ----> 자르다
729 ----> 시험
2 ----> 하다
7 ----> 그
113 ----> 마음
8 ----> 의
792 ----> 깊다
376 ----> 뜻
20 ----> 과
171 ----> 생각
3 ----> 을
4613 ----> 알아내다
```

To be better…

⇒ 수정된 모델로 학습중

```
Epoch 2 Batch 0 Loss 3.4638
Epoch 2 Batch 100 Loss 3.7334
Epoch 2 Batch 200 Loss 3.7081
Epoch 2 Batch 300 Loss 3.2587
Epoch 2 Batch 400 Loss 2.8855
Epoch 2 Batch 500 Loss 3.3928
Epoch 2 Batch 600 Loss 3.2957
Epoch 2 Batch 700 Loss 3.4766
Epoch 2 Loss 3.6083
Time taken for 1 epoch 1036.0396175384521 sec

Epoch 3 Batch 0 Loss 3.0938
Epoch 3 Batch 100 Loss 3.0393
```

Short Excuse

⇒ 믿었던 GCP에서 GPU를 인식 못함

⇒ VM 재설치 후 학습 예정

```
-> 2883         **flat_structure(self))
   2884     super(TakeDataset, self).__init__(input_dataset, variant_tensor)
   2885

~/.local/lib/python3.6/site-packages/tensorflow/python/ops/gen_dataset_ops.py in
take_dataset(input_dataset, count, output_types, output_shapes, name)
   5345       else:
   5346         message = e.message
-> 5347       _six.raise_from(_core._status_to_exception(e.code, message), None)
   5348   # Add nodes to the TensorFlow graph.
   5349   if not isinstance(output_types, (list, tuple)):

~/.local/lib/python3.6/site-packages/six.py in raise_from(value, from_value)

NotFoundError: No registered 'TakeDataset' OpKernel for XLA_GPU devices compatibl
e with node {{node TakeDataset}}
         .  Registered:  device='CPU'
 [Op:TakeDataset]
```

```
In [ ]: def evaluate(sentence):
          attention_plot = np.zeros((max_length_targ, max_length_inp))
```

Thank You.