Transformer를 사용한 이미지 캡셔닝 및 비디오 캡셔닝

김기덕O, 이근후*
O(주)쓰리아이퓨처,
*(주)쓰리아이퓨처
e-mail: kimsjpk@naver.comO, lghoo@naver.com*

Image captioning and video captioning using Transformer

Gi-Duk KimO, Geun-Hoo Lee*
O3Ifuture,
*3Ifuture

● 요 약 ●

본 논문에서는 트랜스포머를 사용한 이미지 캡셔닝 방법과 비디오 캡셔닝 방법을 제안한다. 트랜스포머의 입력으로 사전 학습된 이미지 클래스 분류모델을 거쳐 추출된 특징을 트랜스포머의 입력으로 넣고 인코더-디코더를 통해 이미지와 비디오의 캡션을 출력한다. 이미지 캡셔닝의 경우 한글 데이터 세트를 학습하여 한글 캡션을 출력하도록 학습하였으며 비디오 캡셔닝의 경우 MSVD 데이터 세트를 학습하여 학습 후 출력 캡션의 성능을 다른 비디오 캡셔닝 모델의 성능과 비교하였다. 비디오 캡셔닝에서 성능향상을 위해 트랜스포머의 디코더를 변형한 GPT-2를 사용하였을 때 BLEU-1 점수가 트랜스포머의 경우 0.62, GPT-2의 경우 0.80으로 성능이 향상됨을 확인하였다

키워드: 트랜스포머(transformer), 이미지 캡셔닝(image captioning), 비디오 캡셔닝(video captioning)

I. Introduction

이미지 캡셔닝의 경우 CNN(Convolution Neural Network)와 RNN(Recurrent Neural Network)를 결합하여 이미지에 대한 설명문을 출력해 준다. 이를 적용해 LG유플러스에서 시각장애인용 애플리케이션 설리번 + 앱을 공개하였다. NHIS(National Health Interview Survey)에서 2014년 기준 전 세계에 2억 8,500만 명의 시각장애인이 있으며 UN 발표의 경우 2020년 60세 이상의 인구가 전 인구의 22%를 차지한다고 한다. 그리고 설리번 +앱과 같은 시각 및 독서 보조기는 2026년 33억 90만 달러에 이를 것으로 전망하였다. 비디오 캡셔닝의 경우 시각장애인 보조 기구뿐만 아니라 자연어 입력의 동영상 검색에 사용할 수 있다. 본 논문에서는 이미지의 경우 AIHUB에서 공개한 COCO Dataset[14] 한글 캡션을 학습하여 한국어 캡션을 출력하는 방법을 제안한다. Okt 라이브러리를 사용하여 한글 캡션을 형태소 분리하고 한글 형태소를 토크나이징 하는 방법을 적용하여 언어 모델을 만들고 학습을 진행하였다. 비디오 캡셔닝의 경우 사전 학습된 CNN 중간계층 출력을 통해 영상 픽셀을 임베딩하고 트랜스포머의 인코다-디코더를 거쳐 캡션을 출력하였다. 그리고 출력 캡션의 성능을 높이기 위해 트랜스포머의 디코더를 GPT-2[1]로 바꾸어 학습하였다.

본 논문에서 2절에 이미지 캡셔닝과 비디오 캡셔닝의 관련 연구에 관해서 기술하고 3절에 논문에서 제안한 모델에 대한 설명을 기술한다. 마지막으로 4절에 결론을 기술한다.

II. Preliminaries

1. Related works

1.1 이미지 캡셔닝

딥러닝을 사용한 이미지 캡셔닝의 경우 Show and tell[2]에서 사전 학습된 CNN 중간계층 출력을 통해 이미지 특징을 추출하고 이를 RNN 모델에 입력하여 캡션을 출력하도록 학습하는 방법을 제안하였다. 이후 RNN에 Attention을 적용한 Show attend and tell[3], 객체 검출을 통해 이미지에서 특징을 뽑아내어 성능을 높인 방법[4]과 트랜스포머 모델인 BERT[5], GPT-2를 사용하여 성능을 높인 방법[11]이 제안되었다.

1.2 비디오 캡셔닝

비디오 캡셔닝은 S2VT[6]에서 seq2seq[15] 모델의 인코더에 사전 학습된 CNN 모델의 중간계층 특징을 넣고 디코더에서 캡션을 출력하는 방법이 제안되었다. 이후 트랜스포머를 사용한 방법[7]이 제안되었다.

1.3 트랜스포머

RNN의 경우 시계열 데이터를 처리하기 위해 고안된 모델로 자연어 처리 등 여러 분야에 사용되어 왔으나 긴 대화 등의 장기 의존성 문제가 발생한다. 트랜스포머의 경우 기계번역을 위해 고안된 모델로 MLP(Multi-Layer-Perceptron)와 Attention 방법만을 사용한 모델이다. Attention 방법을 사용해서 CNN이 가지고 있는 지역 정보 문제와 RNN이 가지고 있는 장기 의존성 문제를 해결하였다. 트랜스포머의 인코더, 디코더 레이어를 이용하고 대용량 말뭉치에 대해 사전 학습한 모델을 사용하여 더욱 성능을 높였다. GPT-2의 경우 WebText라 불리는 40기가의 거대한 말뭉치에 인터넷에서 크롤링한 데이터를 합친 말뭉치를 사용하였다.

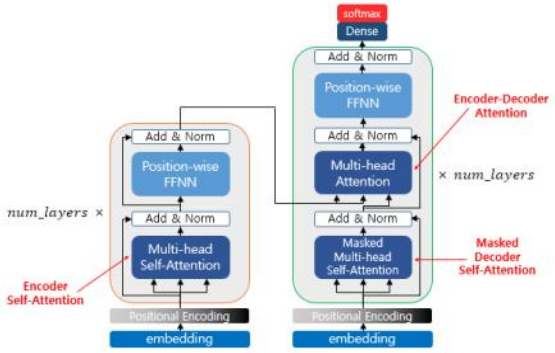

Fig. 1. 트랜스포머 구조 그림

트랜스포머 인코더는 여러 개의 동일한 레이어로 구성되는데 그림 1과 같이 각 레이어의 출력은 다음 레이어의 입력으로 사용된다. 각 레이어는 각 단어 사이의 관계를 계산하는 멀티 헤드 셀프 어텐션과 MLP를 사용하는 순방향 신경망 부분으로 구성된다. 셀프 어텐션은 주어진 query 벡터에 대한 모든 key 벡터와의 점수를 계산하고, 이후 점수를 각각의 value 벡터와 함께 가중합 연산을 통해 인코더에 입력된 벡터들의 최종 표현을 계산한다. 멀티 헤드 셀프 어텐션은 Attention 연산을 여러 개의 가중치 쌍에 대해 계산한 후 여러 개의 결과 벡터를 이어 붙여 사용한다. 이러한 과정을 통해 벡터 간의 다양한 관계에 대한 학습이 가능하다.

III. The Proposed Scheme

본 논문의 이미지 캡셔닝의 경우 학습데이터로 AIHUB에서 제공하는 COCO dataset 한글 캡션을 사용하였다. COCO dataset의 경우 train 데이터 83,000장, validation 데이터 41,000장이 있으며 이미지당 5개의 캡션이 달려있다. 캡션에서 Okt 라이브러리를 통해 형태소 분리를 한 후 이를 띄어쓰기 하여 캡션을 저장하였다. 형태소를 토크나이징하여 언어모델을 만들고 트랜스포머 디코더의 입력으로 넣었다. 토크나이징 과정은 그림 2와 같다.

```
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

sentences = [
  'I love my dog',
  'I love my cat',
  'This neighborhood is quiet and nice.',
  'After I settle in, I'll invite you for a coffee.'
]

tokenizer = Tokenizer(num_words = 100)
tokenizer.fit_on_texts(sentences)
word_dic = tokenizer.word_index
print(word_dic)

sequences = tokenizer.texts_to_sequences(sentences)
print(sequences)

padded = pad_sequences(sequences)
print(padded)

-------------------------------------------------------------------------

{'i': 1, 'love': 2, 'my': 3, 'dog': 4, 'cat': 5, 'this': 6, 'neighborhood': 7, 'is': 8, 'quiet': 9, 'an
d': 10, 'nice': 11, 'after': 12, 'settle': 13, 'in': 14, 'l': 15, 'll': 16, 'invite': 17, 'you': 18, 'for': 1
9, 'a': 20, 'coffee': 21}
[[1, 2, 3, 4], [1, 2, 3, 5], [6, 7, 8, 9, 10, 11], [12, 1, 13, 14, 15, 16, 17, 18, 19, 20]]
[[ 0  0  0  0  0  0  1  2  3  4]
 [ 0  0  0  0  0  0  1  2  3  5]
 [ 0  0  0  0  6  7  8  9 10 11]
 [12  1 13 14 15 16 17 18 19 20]]
```

Fig. 2. 토크나이징 과정 그림

캡션을 입력으로 넣고 빈도수가 높은 단어의 사전을 만든다. 그 후 문장을 단어의 리스트에 따라 순서 시퀀스로 바꾸고 최소 단어 길이만큼 숫자 0을 사용해서 같은 크기의 시퀀스로 변환한다. 트랜스포머 인코더에는 EfficientNet[8]-B0에서 중간계층을 출력으로 하여 이미지 픽셀을 1,280개의 특징으로 임베딩 된 데이터를 입력으로 넣었다. 학습에 사용한 코드는 블로그(https://blog.naver.com/kimsjpk/222571578723)에 올려두었다.

Fig. 3. COCO dataset 이미지

그림 3의 출력 캡션은 "개가 잔디밭에서 프리스비를 잡고 있다"이다.

비디오 캡셔닝은 사전 학습된 InceptionV3[9]를 사용하여 2,048개의 특징으로 임베딩을 하고 트랜스포머 인코더의 입력으로 넣었다. 데이터 세트는 MSVD[10] 데이터 세트를 사용하였다. MSVD의 경우 1970개의 영상으로 구성되어 있고 학습데이터와 검증데이터는 9:1의 비율을 사용하였다. 그리고 비디오 캡셔닝의 성능을 향상하기 위해 디코더를 GPT-2로 바꾸어서 학습하고 두 모델의 성능을 비교하였다. GPT-2를 적용한 모델은 ClipCap[11]를 참고하여 입력으로 CLIP[12]을 적용하고 중간 특징을 뽑아내어 768 크기로 임베딩하고 MLP를 거치고 GPT-2를 디코더로 사용하였다. 학습 코드는 블로그(https://blog.naver.com/kimsjpk/222906154061)에 올려두었다. 성능은 표 1과 같다.

Table 1. 비디오 캡셔닝 모델 간 성능비교

알고리즘	BLEU-1,2,3,4 score			
S2VT	0.755	0.626	0.518	0.409
RecNet[13]	-	-	-	0.499
Transformer 적용	0.620	0.414	0.205	0.086
GPT-2 적용	0.809	0.675	0.546	0.422

알고리즘	METEOR	ROUGE-L	CIDEr
S2VT	0.317	0.673	0.648
RecNet	0.341	0.698	0.803
Transformer 적용	0.217	0.485	0.016
GPT-2 적용	0.364	0.713	0.902

IV. Conclusions

본 논문에서 트랜스포머를 사용한 이미지 캡셔닝 방법과 비디오 캡셔닝 방법을 제안하고 있다. 한글을 형태소 분리하여 토크나이징 방법으로 기존의 이미지 캡셔닝 모델에도 큰 수정 없이 적용 및 학습할 방법을 제안하고 비디오 캡셔닝의 경우 트랜스포머 모델을 GPT-2로 바꿈으로써 더욱 성능이 향상됨을 보였다. 앞으로 본 논문에서 제안한 방법과 함께 다른 트랜스포머 관련 알고리즘을 적용하여 성능을 더욱 높일 수 있을 것이다.

REFERENCES

[1] RADFORD, Alec, et al. Language models are unsupervised multitask learners. OpenAI blog, 2019, 1.8: 9.

[2] VINYALS, Oriol, et al. Show and tell: A neural image caption generator. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. p. 3156-3164.

[3] XU, Kelvin, et al. Show, attend and tell: Neural image caption generation with visual attention. In: International conference on machine learning. PMLR, 2015. p. 2048-2057.

[4] LU, Jiasen, et al. Neural baby talk. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. p. 7219-7228.

[5] DEVLIN, Jacob, et al. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[6] VENUGOPALAN, Subhashini, et al. Sequence to sequence-video to text. In: Proceedings of the IEEE international conference on computer vision. 2015. p. 4534-4542.

[7] IASHIN, Vladimir; RAHTU, Esa. A better use of audio-visual cues: Dense video captioning with bi-modal transformer. arXiv preprint arXiv:2005.08271, 2020.

[8] TAN, Mingxing; LE, Quoc. Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. PMLR, 2019. p. 6105-6114.

[9] SZEGEDY, Christian, et al. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 2818-2826.

[10] CHEN, David; DOLAN, William B. Collecting highly parallel data for paraphrase evaluation. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies. 2011. p. 190-200.

[11] MOKADY, Ron; HERTZ, Amir; BERMANO, Amit H. Clipcap: Clip prefix for image captioning. arXiv preprint arXiv:2111.09734, 2021.

[12] RADFORD, Alec, et al. Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning. PMLR, 2021. p. 8748-8763.

[13] WANG, Bairui, et al. Reconstruction network for video captioning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. p. 7622-7631.

[14] LIN, Tsung-Yi, et al. Microsoft coco: Common objects in context. In: European conference on computer vision. Springer, Cham, 2014. p. 740-755.

[15] SUTSKEVER, Ilya; VINYALS, Oriol; LE, Quoc V. Sequence to sequence learning with neural networks. Advances in neural information processing systems, 2014, 27.