한국어 이미지 캡셔닝 모델 구현

김민재 ○ 김수빈 ○ 진회영
경희대학교 산업경영공학과
경희대학교 소프트웨어융합학과
kmj5596@khu.ac.kr, ksb3966@khu.ac.kr, wlsghldud@khu.ac.kr

Implementation of Korean Image Captioning Model

Minjae Kim ○ SuBeen Kim ○ Hoeyeong Jin
Department of Industrial & Management Systems Engineering, KyungHee University
Department of Software Convergence, KyungHee University

요 약

이미지 캡셔닝이란 이미지가 지닌 정보를 설명해주는 가장 적합한 문장을 생성하는 것이다. 이러한 작업은 컴퓨터 비전과 자연어 처리라는 인공지능 주요 두 분야를 연결하기 때문에 중요한 의의를 갖는 연구이다. 기존에 수행된 연구는 주로 이미지를 영어로만 문장을 생성해냈다면, 우리는 한국어로 문장을 생성하는 알고리즘을 제안한다. Microsoft의 COCO 이미지와 AI HUB의 기계번역이 완료된 설명 데이터를 활용하여 알고리즘을 학습한 후 평가지표로 BLEU를 사용한다. 실험 결과는 우리의 알고리즘을 사용하여 한국어 문장 생성의 발전 가능성이 있음을 보여준다.

1. 서 론

이미지를 문장으로 설명하기 위한 알고리즘들은 많이 연구되었다. 대부분의 연구에서는 알고리즘을 통해 기계번역된 결과와 사람이 직접 번역한 결과가 얼마나 유사한지에 대한 성능을 평가하기 위해 BLEU(Bilingual Evaluation Understudy) Score를 사용한다. 기존 연구에서는 영어 기반 문장의 BLEU 성능 변화에 초점을 맞췄다면, 본 연구에서는 한국어 기반 문장에 대한 BLEU의 성능 변화를 살펴본다.

본 연구의 구성은 다음과 같다. 2 장에서는 프로젝트 구성 및 세부 사항을 설명하고, 3 장에서는 알고리즘에 대한 성능과 문제점을 분석한다. 마지막으로 4 장에서는 결론을 맺는다.

2. 방 법

2.1 데이터 처리

Image Dataset은 Microsoft가 제공하는 MS COCO2014[1]를 사용하였다. 데이터 구성은 train(82,783 개)과 validation(40,504 개)으로, 이미지를 포함하여 각 이미지 당 약 5 개의 Caption이 제공된다. MS COCO2014 train의 80%를 train set, 나머지를 validation set, MSCOCO2014 validation을 test set이 되도록 재구성하였다. 추가로, TensorFlow 프레임워크에서 학습 및 검증을 수행하기 위해서 이미지의 크기를 299 * 299 로, PyTorch 프레임워크에서 학습 및 검증을 수행하기 위해서 이미지의 크기를 224 * 224 로 조절하였다.

MS COCO2014 의 Caption은 영어로 작성되었기에, 한국어로 구성된 Caption은 AI HUB에서 기계번역한 MS COCO 캡셔닝 데이터[2]를 사용하였다. 해당 데이터는 train과 validation이 섞여있기에, train과 validation을 분할하여 캡션 파일을 재구성하였다.

2.2 모델 개발 방법

Image Captioning 분야의 초창기를 이끌었던 'Show and Tell' 논문에 Attention 알고리즘을 적용하여 개선시킨 'Show, Attend and Tell' 논문을 기반으로 학습하였다. 해당 논문에서는 이미지를 Encoding하고 Object Detection을 하는 CNN 모델로 VGGNet을 적용했다. Encoding한 이미지를 단어 단위로 분리하여 전체 이미지에 대한 설명을 생성하는 Decoding 과정에서의 자연어 처리 기법은 RNN 모델로 LSTM(Long-Short Term Memory)를 사용했다.[3] 우리는 논문보다 높은 성능을 얻고자 Encoding 모델로 InceptionV3 를, Decoding 모델은 동일하게 LSTM을 적용하였다.

이후, Encoding 모델은 ResNet101, Decoding 모델은 SKTBrain에서 제공하는 KoBert로 변경하면서 학습을 진행했다.

2.2.1 InceptionV3(CNN) + LSTM(RNN)

Google Colab Pro+(Tesla P100)의 환경에서 TensorFlow 기반으로 TensorFlow 공식 홈페이지의 이미지 캡셔닝 관련 Code를 참조하여 모델의 학습 및 평가를 수행하였다.[4]

Decoding 과정에서 사용되는 Vocabulary는 caption 데이터에서 나타나는 상위 10,000 개의 단어로 구성하였다. InceptionV3 로 추출되는 attention 벡터는 (64, 2048), Batch Size는 64, Epoch는 100 으로 진행하였다.

2.2.2 ResNet101(CNN) + LSTM(RNN)

2.2.1 에서와 같이 Google Colab Pro+의 환경에서 TensorFlow 기반으로 모델의 학습 및 평가를 수행하였다. 2.2.1 에서와 다른 점은 ResNet101 로 추출되는 attention 벡터는 (100, 2048), Epoch는 20 으로 진행한 점이다. Batch Size는 동일하게 64 로 진행했다.

2.2.3 ResNet101(CNN) + KoBert(RNN)

RTX 3090 의 환경에서 PyTorch 기반으로 모델의 학습 및 평가를 수행하였다.[5]

Decoding을 위한 Vocabulary는 caption 데이터에서 5 회 이상 나타나는 토큰만 포함되도록 구성하였다. 해당 조건으로 생성한 Vocabulary에 포함된 단어 수는 8,123 개였다. 하드웨어 resource의 한계로 인해 Batch Size는 4, Epoch는 2 로 진행하였다.

2.3 모델 성능 측정 방법

모델의 성능 측정 지표는 BLEU를 사용한다. BLEU score의 범위는 0 에서 1 사이의 값을 갖는다. 참조 문장과 추출한 문장의 유사도가 높을수록 1 에 가까운 값을 얻으며, 참조 문장의 수가 많아질수록 BLEU가 높게 나타난다.

하지만, BLEU 지표에는 단점이 존재한다. BLEU를 계산할 때는 참조 문장과 추출한 문장에 포함된 단어가 같아야만 높은 측정 값이 나타난다. 즉, 같은 의미를 갖더라도 단어가 다르다면 틀리게 판단하여 낮은 측정 값이 나타나는 것이다.[6] 하지만, 기계 번역의 유사도를 측정하는 데 BLEU를 대체할 만한 지표가 없기에 성능 측정 방법을 BLEU로 선정하였다.

모델 성능(test set의 BLEU) 외에, 2.2.1 과 2.2.2 에서 train과 validation set에 대한 loss 값의 측정을 위해서 SparseCategoricalCrossentropy 함수를, 2.2 .3 에서 train과 validation set에 대한 loss 값의 측정을 위해서 CrossEntropyLoss 함수를 사용하였다.

3. 결 과

3.1 InceptionV3 + LSTM

train과 validation을 100 Epoch를 수행한 결과, train loss는 0.4, validation loss는 0 에 근접한 값이 도출되었다.

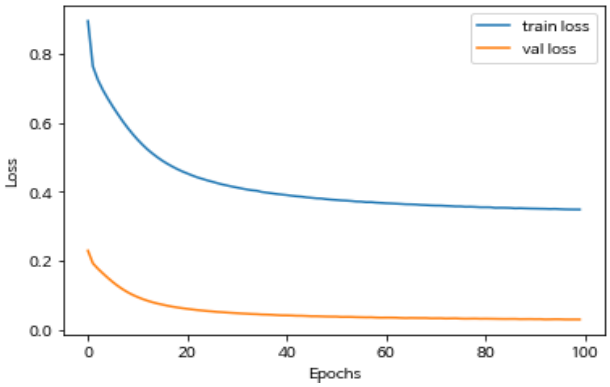

그림 1] InceptionV3 + LSTM의 Train & Validation Loss

test set에 대한 BLEU 지표는 다음과 같이 나타났다.

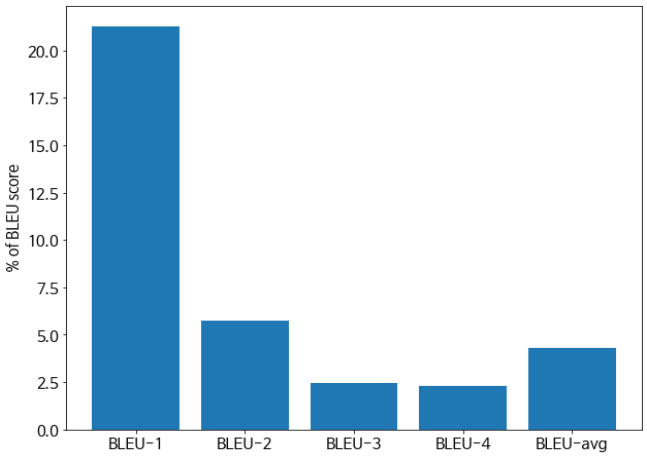

그림 2] InceptionV3 + LSTM의 Test set BLEU score

BLEU-1 은 0.213, BLEU-2 는 0.057, BLEU-3 은 0.025, BLEU-4 는 0.023, 평균 BLEU는 0.043 이다. 해당 모델을 사용하여 특정 이미지의 캡셔닝을 수행한 결과는 다음과 같다.

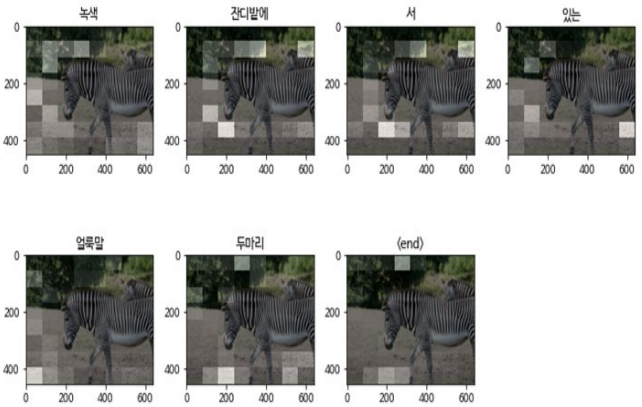

그림 3] InceptionV3 + LSTM의 캡션 예측 example

참조 문장은 '얼룩말 두마리가 몇몇 식물들 옆의 들판을 걷고 있다'이다. 그리고 출력 문장은 '녹색 잔디밭에 서 있는 얼룩말 두마리'이다. 이 예제에 대한 평균 BLEU는 0.028 이다.

3.2 ResNet-101 + LSTM

train과 validation을 20 Epoch를 수행한 결과, 3.1 과 유사하게 train loss는 0.4, validation loss는 0 에 근접한 값이 도출되었다.

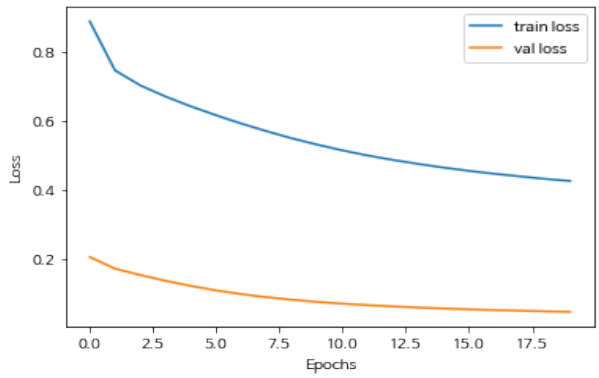

그림 4] ResNet101 + LSTM의 Train & Validation Loss

test set에 대한 BLEU 지표는 다음과 같이 나타났다.

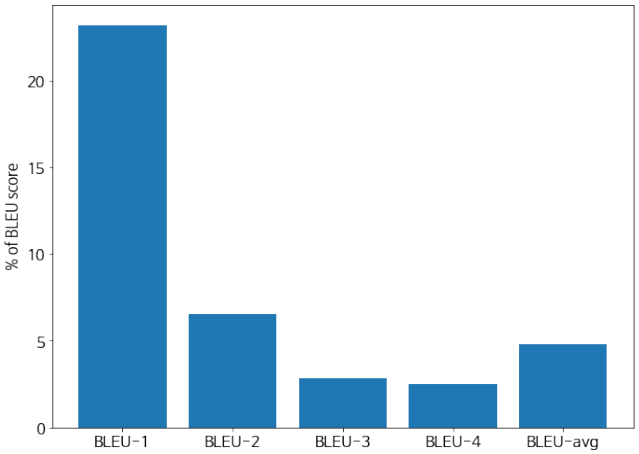

그림 5] ResNet101 + LSTM의 Test set BLEU score

BLEU-1 은 0.232, BLEU-2 는 0.066, BLEU-3 은 0.029, BLEU-4 는 0.025, 평균 BLEU는 0.048 이었다. 해당 모델을 사용하여 특정 이미지의 캡셔닝을 수행한 결과는 다음과 같다.

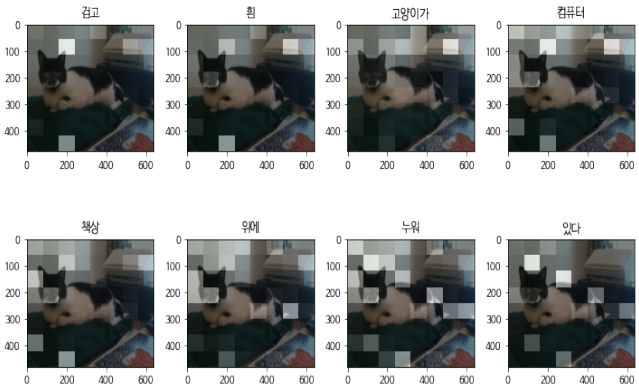

그림 6] ResNet101 + LSTM의 캡션 예측 example

참조 문장은 '검은 색과 흰색의 고양이가 녹색 베개 위에 누워 있다', 출력 문장은 '검고 흰 고양이가 컴퓨터 책상 위에 누워 있다'이다. 이 예제에 대한 평균 BLEU는 0.128 이다.

3.3 ResNet101 + KoBert

train과 validation을 2 Epoch를 수행한 결과, train loss는 7.3, validation loss는 7.1 으로 도출되었다. 이를 통해 해당 모델이 데이터를 제대로 학습하지 못했음을 알 수 있었다.

test set에 대한 BLEU 지표는 다음과 같이 나타났다. BLEU-1 은 0.122, BLEU-2 는 0.014, BLEU-3 은 0.006, BLEU-4 는 1.25e-308, 평균 BLEU는 5.95e-79 이었다. 해당 모델을 사용하여 특정 이미지의 캡셔닝을 수행한 결과는 다음과 같다.

[그림 7] ResNet101 + KoBert 캡션 예측 example

참조 문장은 '모 <unk> 머리를 한 새 한마리가 접시 잔디밭에 서 있습니다', 출력 문장은 '한 <unk> <unk> 있는 있는 있는'이다. 이 예제에 대한 평균 BLEU는 0.024 이다. 참조 문장에서 <unk>가 나온 것은 PyTorch 환경에서 RNN을 KoBert로 변경 한 Vocabulary를 구성하는 과정에서 문제가 발생한 것으로 판단된다.

[표 1] 모델별 훈련 시간 및 Parameter 수

Model	Training Time per Epoch	Total Time	Number of Parameter
InceptionV3 + LSTM	15 mins	24H	23.83M
ResNet-101 + LSTM	10 mins	4H	44.5M
ResNet-101 + KoBert	4 days	1week	154.5M

InceptionV3 에서 ResNet-101 로 CNN을 변경한 후, Epoch 수를 20%로 줄였음에도 성능 향상 및 훈련 시간을 단축하였다. 그러면서, 한국어 문장 생성에 적합하도록 RNN을 KoBert로 변경하였지만 성능 저하 및 훈련 시간 증가 현상이 발생했다.

4. 결 론

Inception V3 에서 ResNet-101 로 모델을 변경했을 때, 더 적은 훈련시간과 Epoch 수만으로 동일한 성능의 Captioning 결과를 얻어낸 것은 긍정적인 부분이다. 다만, KoBert를 이용한 분석에서 성능 저하 및 훈련 시간 증가 문제에 대해서는 추후 연구가 필요한 부분이다.

본 연구에서는 CNN과 RNN의 알고리즘에 따라 BLEU의

성능 변화에 대해 분석하였다. 또한 분석한 결과를 토대로 한국어 Image Captioning에 적합한 최적의 알고리즘을 결정하기 위한 연구를 진행했다.

향후 연구로는 Decoder Model의 작동되는 과정에 대한 추가적인 이해를 바탕으로 KoBert, Bert 등의 한국어에 적합한 이미지 캡셔닝 모델 알고리즘 구현에 관한 연구를 수행할 예정이다.

해당 연구를 사진으로부터 추출한 문장을 TTS(Text to Speech)나 점자 출력 등으로 발전시킴으로써 시각 자료에 대해 정보 제약이 큰 시각장애인들에게 이미지의 정보를 시각 외의 방법으로 인식하는 데 도움을 줄 수 있다.

참고 문헌

[1] https://cocodataset.org/#home

[2] https://aihub.or.kr/keti_data_board/visual_intelligence

[3] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel and Yoshua Bengio, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, arXiv: 1502.03044, 2016.

[4] https://www.tensorflow.org/tutorials/text/image_captioning

[5] https://github.com/ajamjoom/Image-Captions

[6] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, “BLEU: A method for automatic evaluation of machine translation”, ACL, 2002.