creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

조선대학교
CHOSUN UNIVERSITY

[UCI]I804:24011-200000279330

2020년 2월

석사학위논문

Self-Attention GAN을 이용한 문장으로부터 이미지 생성 연구

조선대학교 대학원

컴퓨터공학과

노 주 현

조선대학교
CHOSUN UNIVERSITY

Self-Attention GAN을 이용한 문장으로부터 이미지 생성 연구

Text to image by Self-Attention
Generative Adversarial Network

2020년 2월 25일

조선대학교 대학원

컴퓨터공학과

노 주 현

조선대학교
CHOSUN UNIVERSITY

Self-Attention GAN을 이용한 문장으로부터 이미지 생성 연구

지도교수 김 판 구

이 논문을 공학석사학위신청 논문으로 제출함

2019년 10월

조선대학교 대학원

컴퓨터공학과

노 주 현

조선대학교
CHOSUN UNIVERSITY

노주현의 석사학위논문을 인준함

위원장 조선대학교 교 수 정 일 용 (인)

위 원 조선대학교 교 수 정 호 엽 (인)

위 원 조선대학교 교 수 김 판 구 (인)

2019년 11 월

조선대학교 대학원

조선대학교
CHOSUN UNIVERSITY

목 차

ABSTRACT

Ⅰ. 서 론 ····· 1
A. 연구의 배경 및 목적 ····· 1
B. 연구 내용 ····· 2

Ⅱ. 관련연구 ····· 3
A. seq2seq ····· 3
B. Bidirectional LSTM ····· 5
C. Transformer ····· 6
D. Scaled dot-product attention ····· 9
E. Generative adversarial network ····· 12

Ⅲ. Text를 이용한 이미지 생성 ····· 14
A. Embedding ····· 14
B. Generator ····· 18
C. Discriminator ····· 20
D. Loss ····· 22
E. Training ····· 23

Ⅳ. 실험 및 결과 분석 ····· 24
A. 실험 환경 ····· 24
B. 실험 결과 분석 ····· 26

Ⅴ. 결 론 ····· 29

조선대학교
CHOSUN UNIVERSITY

참고문헌 ····· 30

조선대학교
CHOSUN UNIVERSITY

표 목 차

표 1. 실험 환경 및 설계 ····· 24
표 2. Inception Score ····· 26

조선대학교
CHOSUN UNIVERSITY

그 림 목 차

그림 1. Seq2Seq의 구조 ····· 3
그림 2. Seq2Seq의 Encoder ····· 4
그림 3. Seq2Seq의 Decoder ····· 4
그림 4. Bidirectional LSTM의 구조 ····· 5
그림 5. Transformer의 Encoder ····· 7
그림 6. Transformer의 입력구조 ····· 7
그림 7. positional Encoding의 예시 ····· 8
그림 8. Q, K, V 벡터를 만드는 과정 ····· 9
그림 9. Attention score를 얻어내는 예시 ····· 10
그림 10. 최종적인 Attention Value 추출 ····· 11
그림 11. Generative Advesarial Network ····· 12
그림 12. Bidirectional LSTM ····· 15
그림 13. Attention Score ····· 16
그림 14. Generator의 구조 ····· 18
그림 15. Upsampling의 원리 ····· 19
그림 16. Discriminator의 구조 ····· 20
그림 17. Downsampling의 원리 ····· 21
그림 18. Concatenate embedding&downsampling ····· 21
그림 19. 이미지와 텍스트 데이터 셋 ····· 25
그림 20. <start>토큰과 <end>토큰을 포함한 텍스트 예시 ····· 25
그림 21. 단어를 숫자로 바꾸는 예시 ····· 25
그림 22. Self-Attention GAN의 실험 결과 1 ····· 27
그림 23. Self-Attention GAN의 실험 결과 2 ····· 27
그림 24. 설명이 잘 된 경우 생성된 결과 ····· 28

조선대학교
CHOSUN UNIVERSITY

ABSTRACT

Text to image by Self-Attention Generative Adversarial Network

No, Juhyeon
Advisor : Prof. Kim, PhanKu, Ph.D.
Department of Computer Engineering
Graduate School of Chosun University

It is very difficult for computers to understand human language. For machine translation, we have long used statistical methods and recently started using deep learning. However, the translation has low quality and high quality machine translate model hasn't appear. In particular, the biggest problem was the lack of understanding the context in the sentence. A number of studies have recently been conducted to solve these problems and machine translation using RNN has shown results. However, it was still not able to capture contextual meaning and it was difficult to expect an accurate translation due to the problem of RNN. However, with the advent of the Attachment Mechanism, improvements in machine translation have been made, and not long sentences show great performance improvements.

Deep learning, meanwhile, has shown tremendous results in recent years in the image field. Image recognition and classification have already surpassed human beings and have recently shown achievements such as reproducing and painting styles of famous artists.

With the development of these deep learning models, this study proposes self-Attention GAN, a model that uses text to grasp the meaning of the text and make images accordingly.

조선대학교
CHOSUN UNIVERSITY

Ⅰ. 서 론

A. 연구의 배경 및 목적

컴퓨터가 인간의 언어, 즉 자연어를 처리하는 일은 매우 어려운 일이다. 인간의 언어는 컴퓨터가 이해하기에 너무 복잡했고 동시에 인간의 언어를 처리하는 방법들에 한계가 너무 명확했다. 그러나 기술의 발전과 연구자들의 노력으로 기계 번역의 경우 상당한 정확도를 보여주며 컴퓨터가 자연어를 처리할 수 있다는 가능성을 보여주었다. 초창기 기계번역은 규칙 기반 번역으로 개발 과정에서 언어학자가 중심이 되어 구축되었다. 알고리즘의 기초가 문법에 초점이 맞춰져있었기 때문에 정확도가 상당했으나 개발 비용과 시간이 많이 들기 때문에 전문기업들이나 할 수 있는 방법이었다. 오랫동안 통계적 방식을 이용한 방법을 사용해왔었다. 통계 기반 기술은 빅데이터를 이용해 통계적으로 규칙을 생성해 번역하는 방법이다. 그러나 통계적 방식은 최근 딥러닝을 이용한 번역을 사용하기 시작하였다. 그러나 아직까지 번역의 질이 많이 떨어지고 문장을 정확하게 번역하는 기계번역은 나타나지 못하고 있다. 특히 문장에서 문맥을 잘 이해하지 못하는 점이 가장 큰 문제로 나타났다. 최근 이러한 문제점을 해결하고자 하는 여러 가지 연구가 진행 되었고 RNN을 이용한 기계번역이 성과를 나타냈다. 그러나 여전히 문맥적 의미를 담아내지 못했고 RNN의 문제점(기울기 소실)으로 인하여 정확한 번역을 기대하기 힘들었다. 하지만 Attention Mechanism의 등장으로 기계번역에서의 개선을 이루어 냈으며 길지 않은 문장에서는 큰 성능의 발전을 보여주게 된다.

한편 이미지 부분에 있어서 딥러닝은 최근에는 엄청난 성과를 보여주고 있다. 이미지 인식, 분류 등은 이미 인간을 뛰어 넘었으며 최근에는 유명 화가의 화풍을 재현하여 그림을 그려내는 등의 성과를 보여주고 있다.

이러한 딥러닝 모델들의 발전으로 본 연구에서는 텍스트를 이용하여 그 텍스

조선대학교
CHOSUN UNIVERSITY

트의 의미를 파악하여 이에 맞는 이미지를 만드는 모델인 Self-Attention Genrative Text to Image Adversarial Network를 제안한다. 이 모델은 이미지를 설명하는 텍스트를 입력으로 그 설명에 맞는 이미지를 생성해내는 모델이다. Attention Mechanism을 적용하여 텍스트를 임베딩하고 그 결과를 이용하여 이미지를 생성한다. 이때 이미지를 생성하기 위해 Generative Adversarial Network를 이용하여 이미지를 생성해 내며 추출해낸 문장벡터는 기존의 방법인 LSTM(long short term memory)이 아닌 Self-Attention을 적용시킨 LSTM을 이용한다. 때문에 문맥의 의미를 잘 파악할 수 있을 것으로 기대할 수 있다. 문맥의 의미가 잘 파악 된다면 이미지 생성에 있어 각 단어가 더 잘 반영된 결과가 생성되리라 기대할 수 있다.

B. 연구내용

본 논문은 일반적인 워드임베딩이 아닌 문맥의 의미를 파악할 수 있는 Attention-Mechanism을 이용하여 추출된 feature를 이용하여 이미지를 생성하는 연구이다. Attention Mechanism을 적용하기 위해서 벡터화된 단어로 이루어진 문장을 Bidirectional LSTM을 이용하여 입력받고 결과에 Self-Attention을 적용하여 문장에서 핵심적인 단어들을 찾아내 LSTM의 결과에 다시 적용시킨다. 이렇게 만들어진 결과는 문장의 의미를 파악한 문맥을 가진 벡터라고 할 수 있다. 임베딩 Layer를 통과하여 만들어진 문장행렬을 이용하여 GAN은 이미지를 생성해낸다. 이미지를 생성해 내는 Generator는 문장행렬을 Upsampling하는 과정을 거쳐 $64 \times 64 \times 3$ 의 해상도를 가진 이미지를 생성해 낸다. Discriminator는 이미지를 잘 생성해 낼 수 있도록 계속해서 Generator의 결과물을 감시하고 평가하며 Generator가 이미지를 잘 생성해 낼 수 있도록 피드백하게 된다. 이렇게 만들어진 결과는 Inception score를 이용하여 평가하였다.

조선대학교
CHOSUN UNIVERSITY

Ⅱ. 관련연구

A. Seq2Seq

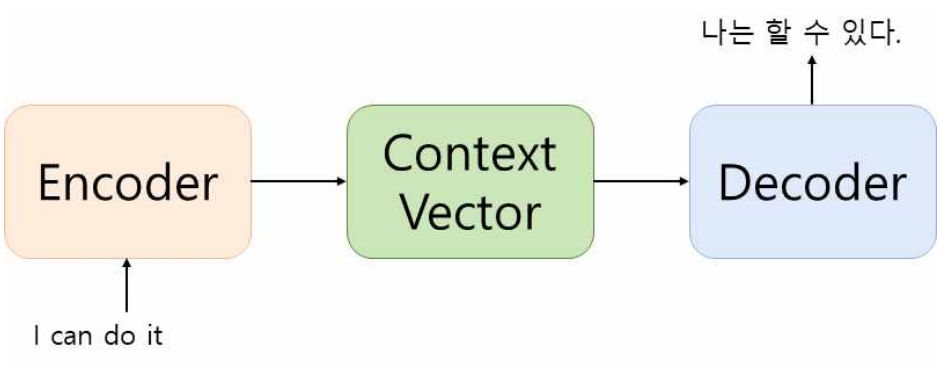

그림 1. Seq2Seq의 구조

신경망의 발전으로 자연어 처리 분야에서는 연속적인 데이터를 처리하기 위하여 RNN이라는 신경망을 사용하게 되었다. 이러한 신경망은 컴퓨터가 인간의 언어를 이해할 수 있게 해주었으며 그 동안 통계방식으로 수행되던 기계번역을 컴퓨터가 수행할 수 있게 해주었다. 이러한 기계번역을 가장 잘 수행하는 모델 중에 Seq2Seq 모델은 RNN을 기반으로 만들어진 모델이다[1]. Seq2Seq는 인코더와 디코더 두 개의 구조로 이루어져 있다. 인코더는 원래의 입력을 처리하여 입력 문장을 컨텍스트 벡터로 만든다.

조선대학교
CHOSUN UNIVERSITY

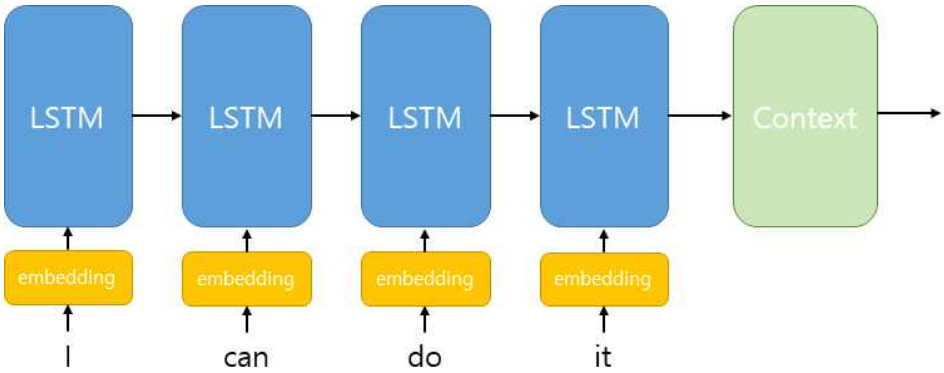

그림 2. Seq2Seq의 Encoder

디코더는 인코더의 마지막 LSTM의 은닉상태를 의미하는 컨텍스트 벡터를 입력으로 사용한다. 디코더는 인코더의 첫 번째 입력을 이용하여 첫 번째 단어를 예측하게 된다. 예측된 단어는 컨텍스트 벡터와 함께 다음 시점의 입력이 되고 결과로 나온 예측을 다시 컨텍스트 벡터와 함께 다음 시점의 입력으로 쓰는 반복적인 작업이 계속된다. 디코더의 입력과 출력은 그림 3처럼 간단하게 표현할 수 있다.

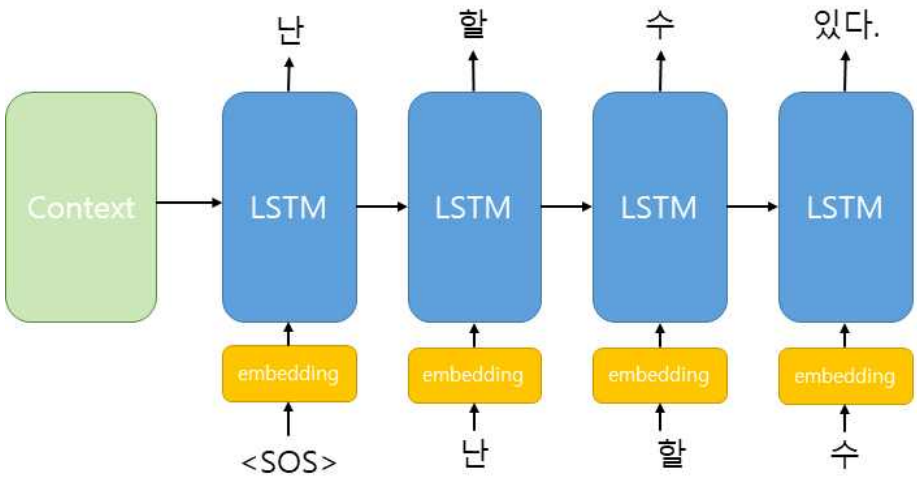

그림 3. Seq2Seq의 Decoder

조선대학교
CHOSUN UNIVERSITY

B. Bidirectional LSTM

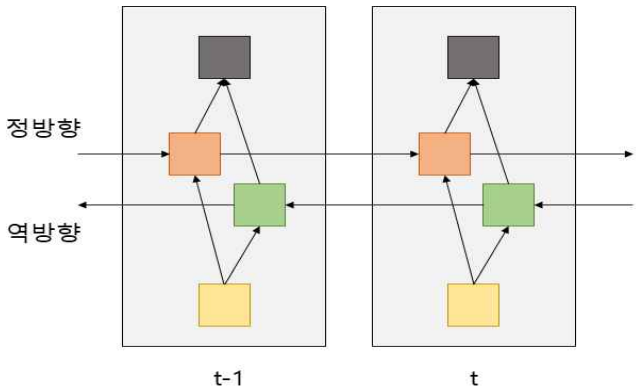

그림 4. Bidirectional LSTM의 구조

기존의 RNN구조는 입력 데이터를 순서대로 처리하기 때문에 정방향으로 추론한다고 할 수 있다. 그러나 정방향 추론뿐만 아니라 역방향 추론 역시 유의미한 결과를 가질 수 있다. 예를 들어 '나는 __을 뒤집어쓰고 펑펑 울었다'라는 문장은 빈칸 뒤의 단어들이 더 중요하다. '나는'이라는 단어보다 '를 뒤집어쓰고 펑펑 울었다'를 보고 더 많은 정보를 알 수 있다. 그런 이유로 정방향뿐만이 아닌 역방향을 동시에 고려하는 RNN구조가 제안 되었다. 이러한 RNN구조를 Bidirectional RNN 혹은 LSTM이라고 부른다. Bidirectional LSTM[3]은 정방향과 역방향을 모두 고려하기 위해 두 종류의 Hidden Unit을 사용한다. 하나는 정방향 LSTM이고 하나는 역방향 LSTM으로써 입력을 각각의 Hidden Unit에 입력하게 된다. 출력은 두가지 Hidden Unit으로부터 값을 전달받아 식 (1)처럼 Concatenate하도록 계산된다.

$$\vec{h} = \overrightarrow{LSTM}(w_t, \overrightarrow{h_{t-1}})$$

$$\overleftarrow{h} = \overleftarrow{LSTM}(w_t, \overleftarrow{h_{t+1}})$$

$$\mathrm{H} = \vec{h} + \overleftarrow{h} \qquad (1)$$

조선대학교
CHOSUN UNIVERSITY

C. Transformer

이전까지 sequence 데이터를 처리하기 위해 쓰였던 RNN 모델의 문제점은 입력이 길어지면 잘 처리하지 못하는 문제가 있었다. 이러한 문제를 해결하기 위해서 앞서 언급한 Attention 메커니즘만을 사용하여 모델을 구성하게 된다[3]. Transformer는 RNN을 사용하는 대신 Multi-head Attention을 사용하는 모델이다. 기존의 attention은 RNN의 출력 벡터를 합쳐 다음 단어의 예측에 사용했지만 Multi-head Attention은 순차적인 입력을 가지는 대신 전체 문장 벡터를 입력받고 이 입력에 Attention을 적용한다. 또한 Attention을 각각 여러 번 적용시키는데 그렇기 때문에 Multi-head를 가진다하여 Multi-head attention이라 불린다. 이전의 모델들과 마찬가지로 Encoder-Decoder 구조를 이용하였으며 특이하게도 인코더와 디코더가 N개로 이루어졌다는 점이다. 이 모델에서 RNN이나 CNN을 사용하지 않고 self-attention을 사용하는데 이러한 구성에는 세 가지의 이점이 있다. 첫 번째는 연산량 감소의 효과이다. RNN을 Attention Layer로 대체함으로써 연산량을 엄청나게 줄일 수 있다. 두 번째는 병렬화가 가능하다는 점이다. 세 번째는 문장이 길어졌을 때 잘 작동하지 않던 점을 쉽게 해결 가능해진다는 장점이 있다. 이러한 방법으로 인해 긴 문장에서도 잘 작동하고 RNN을 사용하지 않음으로써 RNN의 단점인 기울기 소실 문제까지 해결된 모델이라 할 수 있다.

조선대학교
CHOSUN UNIVERSITY

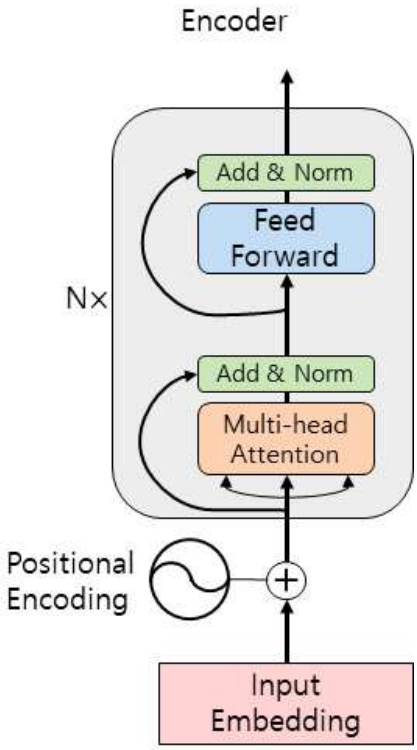

그림 5. Transformer의 Encoder

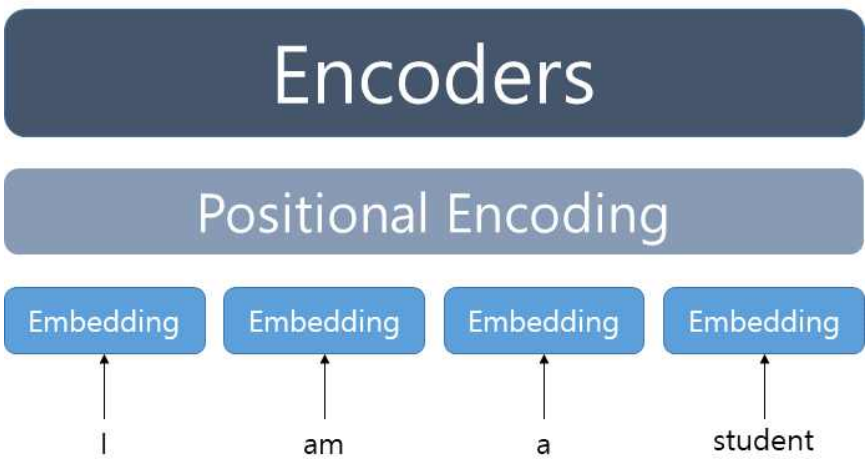

그림 6. Transformer의 입력구조

조선대학교
CHOSUN UNIVERSITY

기존의 자연어 처리에서 좋은 성능을 보였던 RNN이 자연어 처리에서 유리했던 이유는 문장을 단어의 순서대로 입력받는 특성 때문에 각 단어의 위치에 대한 정보를 가지기 때문이었다. 그러나 Transformer는 RNN을 쓰지 않기 때문에 각 단어의 위치 정보를 전달해야할 필요가 생기게 된다. Transformer는 단어의 위치정보 전달을 위하여 포지셔널 인코딩을 이용한다. 그림 5와 6은 입력인 임베딩 벡터들이 인코더의 입력으로 사용되기 전에 각각의 위치정보를 포지셔널 인코더를 이용하여 더해지는 과정을 보여준다. 포지셔널 인코딩은 식 (2)의 함수를 사용하여 값들을 가지게 된다.

$$\begin{aligned} PE_{(pos,\,2i)} &= \sin(pos/10000^{2i/d_{\mathrm{model}}}) \\ PE_{(pos,\,2i+1)} &= \cos(pos/10000^{2i/d_{\mathrm{model}}}) \end{aligned} \qquad (2)$$

식 (2)의 변수 pos는 입력 문장에서의 임베딩 벡터의 위치, i는 임베딩 벡터 내의 차원의 인덱스를 의미하며, d_{model} 변수는 출력차원을 의미한다. 위 식에 따라 임베딩 벡터 내의 차원의 인덱스가 짝수인 경우에는 사인 함수의 값을 사용하고 홀수인 경우에는 코사인 함수의 값을 사용한다.

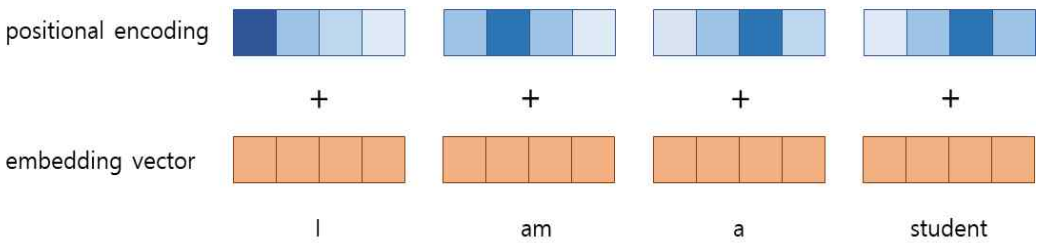

그림 7. positional Encoding의 예시

조선대학교
CHOSUN UNIVERSITY

D. Scaled dot-product Attention

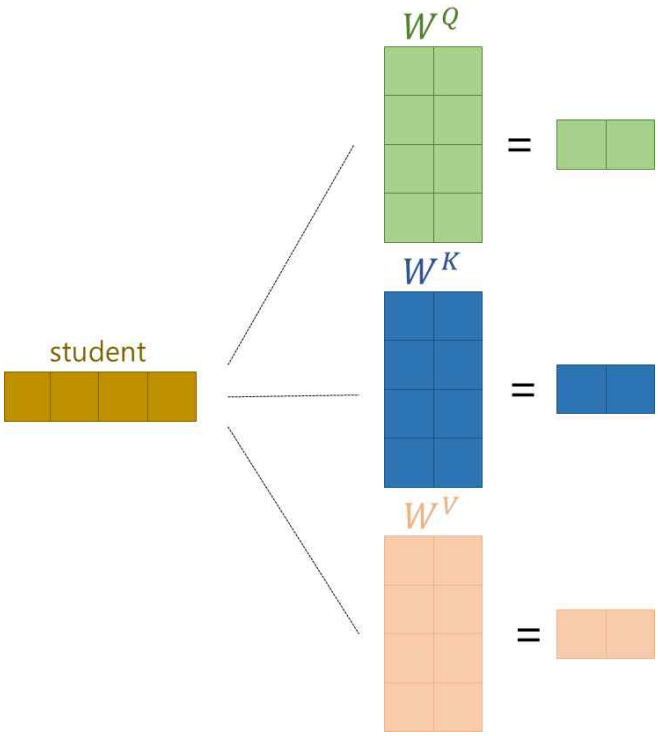

그림 8. Q, K, V 벡터를 만드는 과정

Scale dot-product Attention는 문장을 Query, Key, Value 형태로 바꾸어 수행하게 된다. 먼저 그림 9처럼 Q, K, V 벡터를 구한다. 각각의 벡터는 서로 다른 가중치를 이용하여 구하게 된다. 이 과정은 각 단어 벡터로부터 Q벡터, K벡터, V벡터를 얻는 작업을 수행하게 된다. 각각의 Q, K, V 벡터를 구하기 위해 세 개의 Fully-connected Layer를 이용한다. 만들어진 Layer의 출력은 512차원을 가지며 입력은 세 개의 계층 모두 임베딩 벡터이다. 가중합의 결과를 이용하여 Attention Score를 구하게 되는데 먼저 Q벡터에 대해서 K 벡터와의 유사도를 구한다. 이렇게 구해진 유사도를 Key와 매핑되는 각각의 V에 반영해주고 유사도가 반영된 V벡터들을 모두 더하게 되면 Attention score가 반영된 결과라고 할 수 있는 값이 나오게 된다. 본 논문에는 여러 가지 어텐션 함수가 있지만 Scaled dot-product Attention을 이용한다. 위에서 설명한 dot-product Attention에

조선대학교
CHOSUN UNIVERSITY

Scale을 적용하게 되는데 결과를 0과 1사이의 값으로 압축시켜 overflow 혹은 underflow를 방지해주는 역할을 하게 된다. 최종 Scaled dot product Attention 함수는 식 (3)과 같이 정의된다.

Scaled dot product Attention :

$$score(q,k) = q \cdot k / \sqrt{n} \quad (3)$$

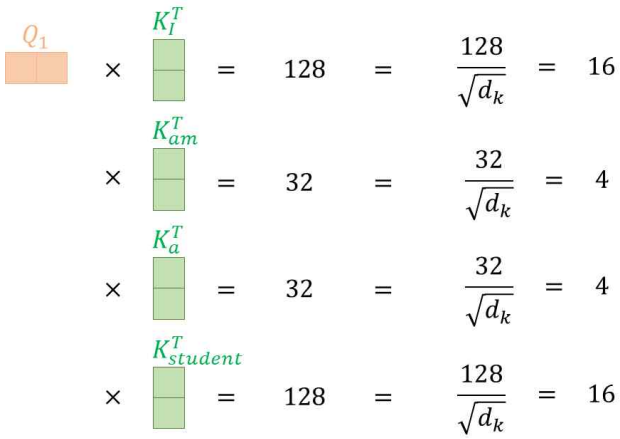

그림 9. Attention score를 얻어내는 예시

그림 9에서 Q와 K의 전치행렬이 곱해진 결과를 Attention score라고 한다. 여기에 K 벡터의 차원인 d_k를 이용하여 스케일을 적용한다. 이렇게 얻은 값을 마지막으로 softmax를 적용하여 Attention distribution을 구하고 이를 V에 곱해 최종적으로 어텐션이 적용된 값을 얻어내게 된다. 이러한 과정은 각각의 단어를 하나씩 적용하는 것이 아닌 행렬을 이용해 일괄적으로 처리하게 된다.

조선대학교
CHOSUN UNIVERSITY

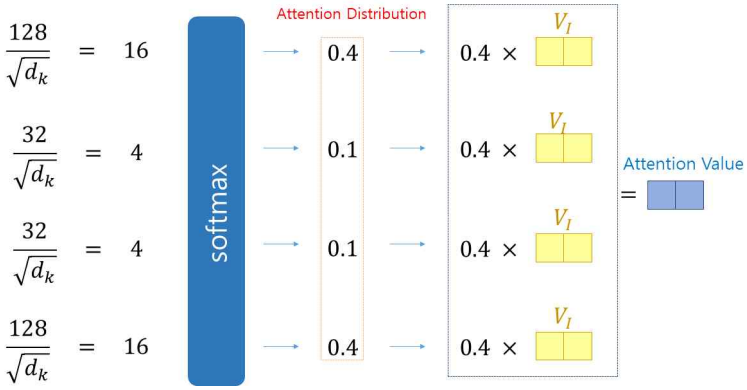

그림 10. 최종적인 Attention Value 추출

조선대학교
CHOSUN UNIVERSITY

E. Generative Adversarial Network

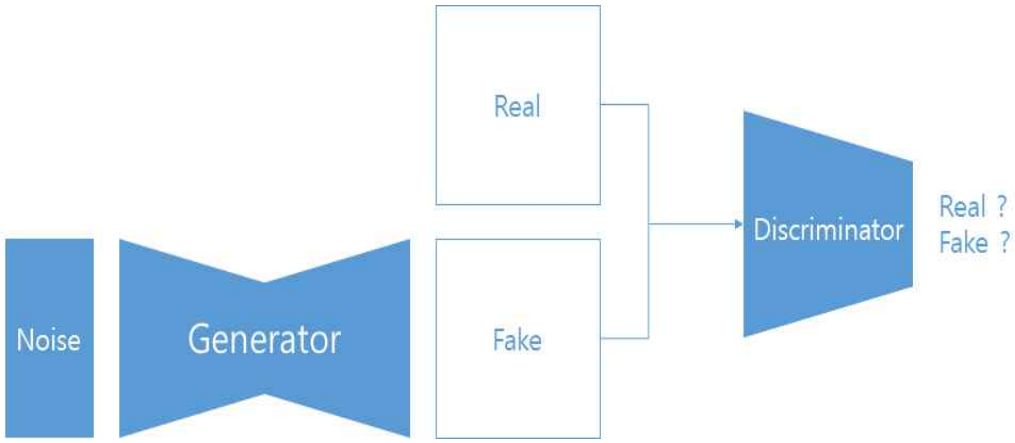

그림 11. Generative Advesarial Network

Generative Adversarial Network(GAN)은 대표적인 비지도 학습 모델로써 대립하는 두 개의 모델이 상호작용하여 이미지를 생성해내는 등의 결과를 생성해 내는 모델이다[6]. GAN의 기본 구조는 그림 3과 같다. Generator는 랜덤 노이즈를 학습하고 이를 바탕으로 실제에 가까운 거짓 데이터를 생성해 낸다. Discriminator는 Generator가 생성해낸 거짓 데이터가 진짜인지 가짜인지 판별하도록 학습한다. 즉, 랜덤 노이즈로부터 Generator가 데이터를 생성해내면 Discriminator는 이를 진짜인지 가짜인지 판별하고 이러한 과정을 반복하여 최종적으로 Generator가 생성한 이미지가 가짜인지 진짜인지 판별하지 못하게 되면 학습이 종료되는 모델이다. 이러한 GAN은 다른 생성 모델들 보다 생성에서 뛰어난 성능을 보여주었으며, 특히 이미지 생성에 있어 사람의 눈으로 실존이미지인지 가상이미지인지 판별하기 어려운 수준까지 발전했다.

$$\min_{G}\max_{D} V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log\ 1 - D(G(z)))] \tag{4}$$

조선대학교
CHOSUN UNIVERSITY

식 (4)에서 G는 Generator, D는 Discriminator이다. 입력 벡터는 x이고 z는 랜덤 노이즈 벡터이다. Discriminator는 큰 값을 가지는 방향으로 훈련되고 Generator는 작은 값은 가지는 방향으로 학습된다.

조선대학교
CHOSUN UNIVERSITY

Ⅲ. Text를 이용한 이미지 생성

A. Embedding

문장을 임베딩하는 모델은 크게 두 가지 부분으로 이루어져있다. 첫 번째는 Bidirectional LSTM을 이용하여 문장을 입력받는 부분, 두 번째는 Bidirectional LSTM의 결과로 나오는 결과를 받아서 Self-Attention을 수행하는 부분이다. 이 결과는 기존의 RNN을 활용한 모델보다 문장에서 각 단어의 의미가 더 정확하게 부여된 벡터라고 할 수 있다.

1. Bidirectional LSTM

입력 문장은 각 단어 벡터의 모임이다. 즉 문장은 단어 벡터의 모임인 행렬로 표현할 수 있다.

$$S=(w_1, w_2, w_3, \dots, w_n) \quad (5)$$

입력 문장 S는 (n, d)형태의 행렬로 표현할 수 있다. 행렬 S를 입력으로 자연어 처리에 가장 많이 이용되는 Bidirectional LSTM을 사용한다. Bidirectional LSTM은 순방향입력과 역방향 입력을 모두 고려한다. 수식으로는 (6)과 같이 표현된다.

$$\begin{aligned} \vec{h} &= \overrightarrow{LSTM}(w_t, \overrightarrow{h_{t-1}}) \\ \overleftarrow{h} &= \overleftarrow{LSTM}(w_t, \overleftarrow{h_{t+1}}) \end{aligned} \quad (6)$$

조선대학교
CHOSUN UNIVERSITY

순방향 입력과 역방향 입력의 hidden state인 각 h를 연결시키면 Bidirectional LSTM의 결과 h_t를 얻을 수 있다. n개의 단어벡터에 모두 적용하여 이 값들을 모아 식 (7)처럼 하나의 행렬 $(n, 2u)$의 크기를 갖는 행렬 M을 만들게 된다.

$$M = \vec{h} + \overleftarrow{h} \qquad (7)$$

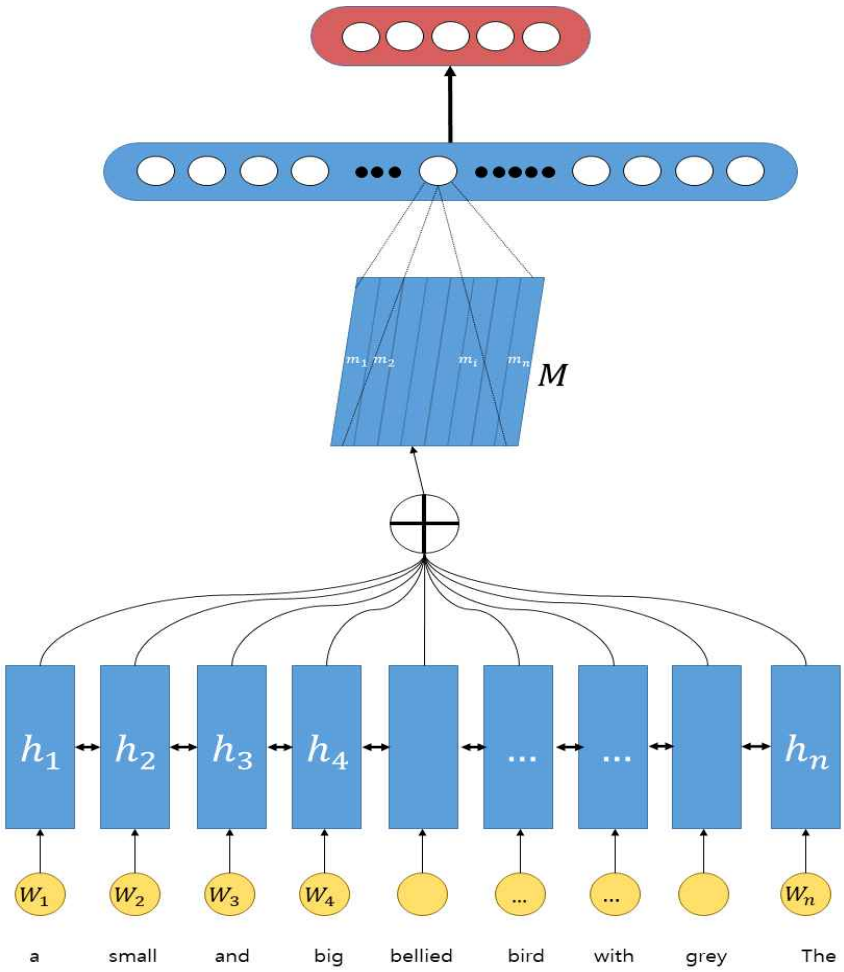

그림 12. Bidirectional LSTM

조선대학교
CHOSUN UNIVERSITY

2. Self-Attention의 적용

Bidirectional LSTM의 결과에 Self-Attention을 적용시키기 위해서 Bidirectional LSTM의 결과 M에 가중치 W_{s1}을 곱한다. W_{s1}은 $(d_a, 2u)$ 크기의 가중치 행렬이다. d_a는 하이퍼파라미터로 적용하고자하는 차원의 크기이다. $2u$는 양방향 LSTM의 차원이다. 활성화 함수로는 tanh를 적용하게 된다. 결과로 나온 (d_a, n) 모양의 행렬에 다시 가중치 w_{s2}를 곱한다. w_{s2}는 d_a크기의 가중치 벡터이다. 이렇게 구해진 값 a_{score}는 단어의 길이와 같은 n차원 벡터이다. a_{score}는 한 문장에 대해서 하나의 의미론적 정보를 담고 있는 값이다. 한 문장은 여러개의 다른 의미들을 가질 수 있다. 이런 점을 반영하기 위해서 w_{s2}를 벡터가 아닌 r크기를 갖는 행렬로 구성하면 여러 의미를 반영할 수 있게 된다. 마지막으로 Softmax를 적용하여 최종적인 attention Score를 구하게 된다. 이 과정은 식 (8)과 같이 표현될 수 있다.

$$A_{score} = softmax(w_{s2}\tanh(w_{s1}M^T)) \qquad (8)$$

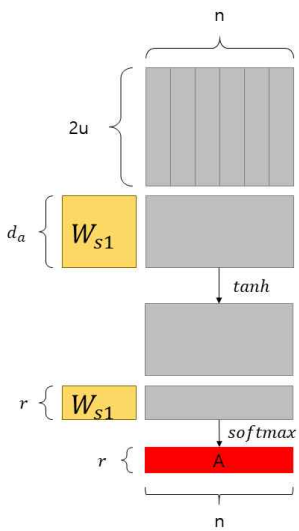

그림 13. Attention Score를 구하는 과정

조선대학교
CHOSUN UNIVERSITY

최종 결과인 A_{score}는 각 토큰들에 얼마나 반영 할지를 결정하는 확률값으로 표현된다. A_{score}를 H에 곱해서 최종적으로 Attention 적용되어 식 (9)의 문장 행렬 M을 구하게 된다.

$$M = A_{score} \times H \qquad (9)$$

마지막으로 임베딩결과를 전달하기 위해 문장행렬 M은 fully-connected Layer를 이용한다. Fully-Connected Layer는 $(n, 2u)$크기를 가지는 문장행렬 M을 reshape하여 $(n \times 2u)$의 모양으로 만들고 $(n \times 2u, E_{\dim})$의 크기를 가지는 가중치에 곱해 최종적으로 $E_{\dim}$크기의 벡터 식 10에 표현된 $X_{embedding}$얻어낸다.

$$X_{embedding} = W_{FC} \times M \qquad (10)$$

앞선 Attention을 수행한 문장행렬의 값들이 비슷한 값들을 가지면 정확한 정보를 전달하기 힘들다. 이러한 문제를 방지하기 위해 문장 행렬의 값들이 비슷한 값을 가지지 못하게 조절해줘야 한다. 따라서 행렬의 값을 조절하기 위해 Penalization으로 Forbenius Norm을 적용하여 문장행렬의 값이 비슷한 값을 가지게 되는 것을 방지한다[5].

$$Penalty = \|(AA^T - I)\|_F^2 \qquad (11)$$

A는 앞서 구한 A_{score}이다. A^T는 A의 전치행렬, I는 단위행렬이다. 구해진 패널티는 loss 함수로써 문장벡터의 값을 조절할 때 사용한다.

조선대학교
CHOSUN UNIVERSITY

B. Generator

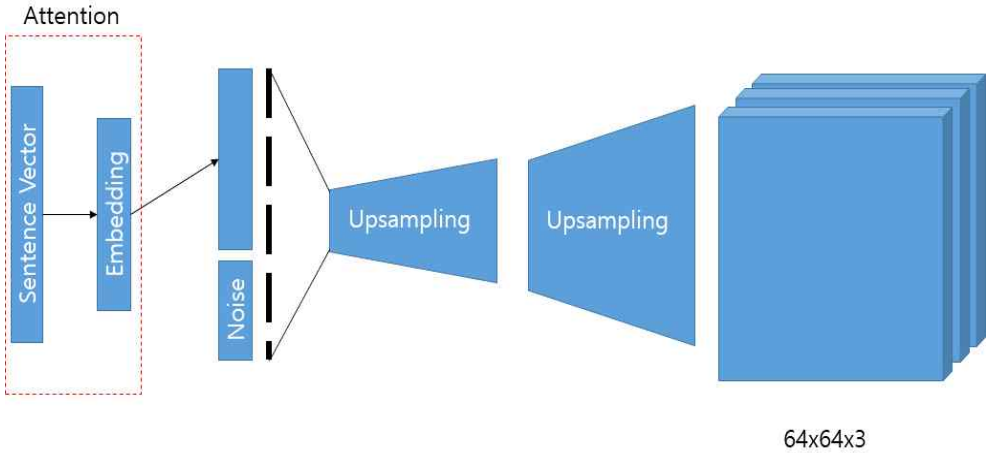

그림 14. Generator의 구조

Generator는 임베딩의 결과를 이용하여 문장에 맞는 이미지를 생성하는 역할을 한다. 그러나 임베딩만을 이용해서 이미지를 만들기에는 $E_{\dim}$만으로는 정보가 부족하기 때문에 랜덤한 노이즈 벡터 Z를 이용한다. Z는 $E_{\dim}$의 차원을 가지도록 설정 했으며 1과 -1사이의 무작위 값들로 초기화된 가우시안 노이즈를 사용했다. Z와 임베딩의 $X_{embedding}$을 Concatenate하여 $E_{\dim} \times 2$ 차원을 가지는 벡터를 생성한다. 이 벡터를 이미지로 만들기 위한 첫 번째 단계로 Fully-Connected Layer를 이용한다. ($E_{\dim} \times 2$, $16 \times 16 \times 128$) 행렬인 Fully-Connected Layer를 통과하여 $16 \times 16 \times 128$의 행렬 $g_{feature}$을 얻는다. 수식으로 표현하면 식 (12)와 같다.

$$g_{features} = W_g \times Concatenate(X_{embedding}, Z_{noise}) \quad (12)$$

조선대학교
CHOSUN UNIVERSITY

$g_{feature}$는 128채널을 가지는 16×16의 이미지이다. Generator의 최종 출력은 $64\times64\times3$이므로 이에 맞추어 Upsampling을 반복 수행한다.

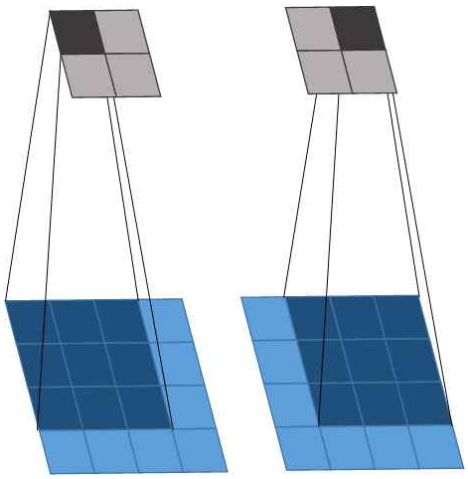

그림 15. Upsampling의 원리

첫 번째 Upsampling에서는 $g_{feature}$를 $32\times32\times64$ 으로 width와 height는 두 배로 커지는 대신 channel의 수를 감소시키며 이미지를 확대한다. 다음은 1×1 convolution을 이용하여 channel을 조정한다. 48개의 채널을 가지도록 조정하였으며 결과는 $32\times32\times48$이 된다. 마지막으로 이미지를 생성해 내기위해서 $64\times64\times3$의 크기를 가지도록 Convolution하여 Generator의 최종결과를 얻게 된다.

조선대학교
CHOSUN UNIVERSITY

C. Discriminator

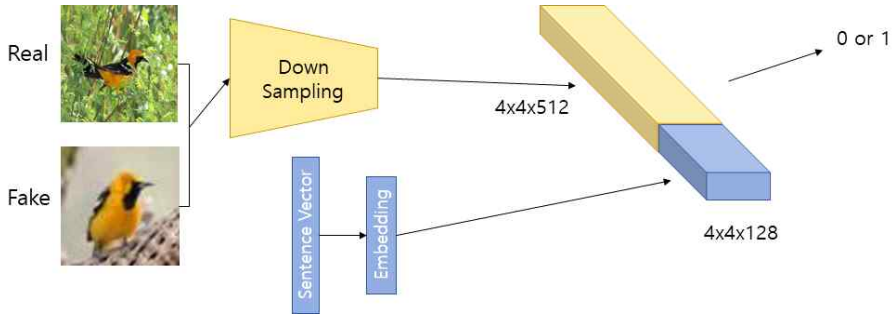

그림 16. Discriminator의 구조

Discriminator는 임베딩의 결과, Generator의 결과, 원래의 이미지를 이용하여 생성된 이미지가 진짜 이미지와 비교하여 얼마나 잘 생성 되었는지를 평가한다. Discriminator의 Input은 [$X_{embedding}$, 이미지]쌍으로 생성된 이미지일 경우는 $[X_{embedding}, G_{img}]$를 이용해서 진짜인지 가짜인지 판별하고 원래의 이미지인 경우에는 $[X_{embedding}, R_{img}]$를 이용해서 진짜인지 가짜인지 판별하게 된다. 가짜 이미지의 경우에는 0에 가깝게 나와야하고 진짜 이미지인 경우에는 1에 가깝게 나오게 하는 것이 Discriminator의 목표라고 할 수 있다. Discriminator는 판별을 위해서 먼저 입력받은 이미지를 Downsampling한다. 입력받은 이미지 $64 \times 64 \times 3$을 downsamping 하여 이미지의 크기는 줄이고 channel의 수를 늘리는 convolution 과정을 거친다. downsampling을 반복하여 최종적으로 $4 \times 4 \times 128$의 크기를 가질 때까지 반복시킨다. 동시에 $X_{embedding}$과 downsampling된 값을 Concatenate하기 위해서 $X_{embedding}$에 $(X_{embedding}, 4 \times 4 \times 128)$의 가중치를 가진 Fully-connected Layer를 적용시킨다. 결과로 얻은 $4 \times 4 \times 128$행렬 $D_{embedding}$의 값을 downsampling의 결과와 concatenate시킨다.

조선대학교
CHOSUN UNIVERSITY

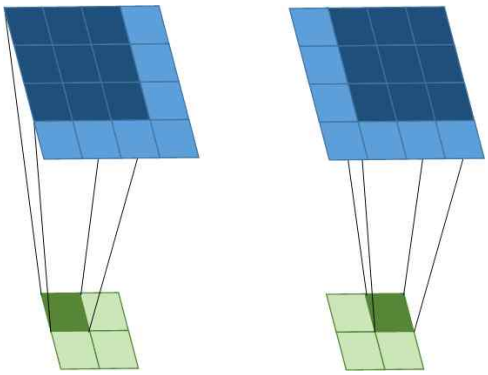

그림 17. Downsampling의 원리

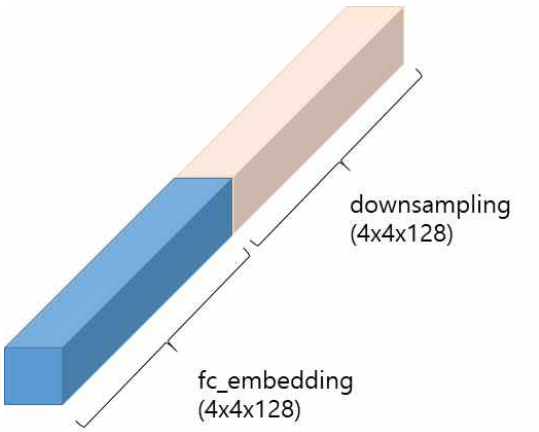

그림 18. Concatenate embedding & downsampling

결과로 나온 $4 \times 4 \times 256$ 행렬을 채널을 1로 하는 Convolution을 수행하여 최종결과로 길이가 $[1]$인 벡터를 얻게 된다. 과정이 복잡해 보이지만 식 (13)과 같이 간단하게 표현할 수 있다.

$$C_{result} = Concatenate(down, fc_{embedding}) \quad (13)$$

조선대학교
CHOSUN UNIVERSITY

D. Loss

Embedding의 학습은 텍스트에 맞는 이미지가 무슨 새인지 분류하는 방식으로 학습이 진행된다. 본 논문에서 사용한 bird 데이터 셋은 200여종의 새가 있고 임베딩은 설명하는 문장을 이용하여 문장이 설명하는 새가 어떤 새인지 분류하는 방향으로 학습이 진행된다. 분류하기 위해서 마지막 층에 softmax를 적용시켰으며 정확도가 95퍼센트를 넘어갈 때까지 학습시킨다.

GAN의 학습은 Discriminator가 Generator가 생성한 가짜 이미지와 원본 이미지를 구분하지 못하게 되는 방향으로 진행된다. Discriminator는 입력받은 문장벡터와 이미지를 최종적으로 진짜인지 가짜인지 판별하는데, Generator가 진짜에 가까운 가짜를 잘 생성해 낼수록 판별하지 못하게 된다. 그렇기 때문에 Generator는 이미지를 잘 생성해내는 방향으로, Discriminator는 잘 구별하는 방향으로 학습이 진행이 된다. 서로 다른 두 개의 네트워크가 학습을 진행하기 때문에 두 개의 네트워크는 서로 다른 손실을 이용하여 학습이 진행하게 된다. Discriminator의 최적화는 문장벡터와 진짜 이미지를 사용하여 나온 출력 y와 가짜 이미지를 사용하여 나온 출력 $\hat{y}$을 이용하여 두 값의 차이를 계산하는 방법으로 이루어진다. 그리고 일반적인 GAN의 Loss에 추가적인 Loss가 추가된다. 각각의 단어에는 수많은 이미지가 매칭 될 수 있다. 이러한 문제점을 해결학기 위해서 임베딩된 문장이 잘못된 이미지를 만들어 내는 경우에 손실률을 늘리는 방법으로 문제를 해결한다[5]. 최종적인 Discriminator의 Loss는 식 (15)와 같다.

$$BCE_x = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(h(x_i;\theta)) + (1-y)\log(1-h(x_i;\theta)) \quad (14)$$

$$Loss_D = BCE_{real} + BCE_{fake} + BCE_{wrong} \quad (15)$$

Generator의 최적화는 Discriminator에 가짜 이미지를 넣은 결과를 이용하여 나온 결과인 식 (16)를 이용한다.

$$Loss_G = BCE_{fake} \quad (16)$$

조선대학교
CHOSUN UNIVERSITY

E. Training

Embedding은 200회 학습이 진행된다. 정확도가 90퍼센트를 달성할 때까지 반복하였으며 이미지를 생성하기 위해서 분류하기 이전의 결과를 생성네트워크에서 사용한다. GAN은 Embedding의 결과를 이용하여 이미지를 생성하기 위해 600번의 학습을 진행한다. 실험의 파라미터는 다음과 같다. 임베딩은 512차원으로 설정한다. LSTM 셀은 양방향을 가지기 때문에 1024의 크기를 갖는다. Embedding의 Fully-Connected Layer는 256, 128의 크기를 가지는 벡터를 결과로 생성하게 하였다. Generator는 Embedding의 결과인 128의 길이를 벡터와 100의 길이를 가지는 noise 벡터를 이용하여 이미지를 생성해낸다. Upsampling은 4회 실시하였고 결과는 $64\times64\times3$의 이미지를 생성하도록 조정하였다. Discriminator는 임베딩의 결과인 128 길이의 벡터를 Fully-Connected layer를 통과시켜 $4\times4\times64$의 길이를 가진 벡터를 얻는다. 또한 입력 이미지는 4회의 downsampling을 거쳐 $4\times4\times512$의 길이를 가진 결과를 얻는다. 이 두 개의 벡터는 Concatenate과정을 거쳐 최종적으로 진짜인지 가짜인지 판별하게 된다.

조선대학교
CHOSUN UNIVERSITY

Ⅳ. 실험 및 결과 분석

A. 실험 환경

표 1. 실험 환경 및 설계

실험 환경	CPU i7-8700K, GPU-1080ti 11gb
트레이닝 셋	75443
발리데이션 셋	18861
테스트 셋	23576
학습 횟수	Embedding 100회, GAN 1000회
학습시간	Embedding 100회당 약 7시간 GAN 100회당 약 23시간

표 1에 나온 것처럼 훈련에는 75443개의 데이터를 사용하였으며 발리데이션은 18861개, 테스트는 23576개의 데이터로 수행하였다. 실험에 사용한 데이터는 CUB_200_2011 데이터로 이미지를 설명하는 텍스트와 이미지 쌍으로 이루어진 데이터 셋이다. 쓰인 데이터 쌍은 그림 18과 같다. 한 개의 이미지 당 열 개의 설명하는 문장이 있기 때문에 하나의 이미지로부터 열 개의 데이터 셋이 생성된다.

조선대학교
CHOSUN UNIVERSITY

this bigger bird has a white belly, breast and head ...
this bird is white and grey in color with a curved ...
the small white and gray bird is standing on rock ...
this bird is short, fat, has webbed feet and the top ...
this bird has a white belly and breast with grey back ...
this larger bird has a yellow bill with pink webbed ...
this bird is very fat with its wings flared out and ...
a white breasted, grey bodied bird with pink, ...
this bird is grey and white it has a yellow beak ...
this medium sized shore bird features a bright ...

그림 19. 이미지와 텍스트 데이터 셋. 하나의 이미지에 열 개의 설명하는 문장이 있다.

<start> this is a green bird with a yellow and white underbelly and white streaks around its eyes. <end>

그림 20. <start>토큰과 <end>토큰을 포함한 텍스트 예시

는 단어는 start 토큰과 end 토큰을 포함하여 1029개의 단어로 이루어져 있다. 각 단어에 숫자를 매칭하여 훈련을 진행하게 된다. 실제로 이러한 단어집을 만드는 tokenizer를 이용하여 문자를 숫자로 바꾸면 그림 20처럼 결과가 나오게 된다.

4	start
11	this
13	is
1	a
53	green
5	bird
8	with
1	a
17	yellow
2	and
12	white
283	
⋮	

그림 21. 단어를 숫자로 바꾸는 예시

조선대학교
CHOSUN UNIVERSITY

B. 실험 결과

표 2. Inception Score[8]

	Dataset	GAN-INT-CLS[6]	Self-Attention GAN
Inceptions core	CUB_200_2011	2.88 ± .04	3.24 ± 0.1

본 논문에서는 실험환경의 제한으로 $64 \times 64 \times 3$의 저해상도 이미지를 생성해 내는 네트워크를 구성하였다. GAN의 성능을 척도하는 방법은 초창기에는 생성된 이미지를 정성적으로 평가하는 방식이 진행 되었으나 Inception Score가 등장하면서 별도의 네트워크를 이용하여 생성된 이미지의 성능을 평가하게 되었다. 성능 평가지표는 Frechet Inception Distance(FID)라고 불리며 Inception network를 사용하여 성능을 측정하는 방법이다. FID는 간단하게 실제 데이터와 가짜 데이터의 feature space상에서의 거리라고 할 수 있다. 진짜 데이터와 가짜 데이터에서 특징들을 추출한 뒤, 두 집합의 특징의 평균과 분산을 구하여 각 값들을 이용하여 거리를 계산하게 된다.

$$FID^2 = \|m_f - m_r\|_2^2 + Tr(C_f + C_r - 2(C_f C_r)^{1/2}) \quad (17)$$

식 (17)의 $(m_r, C_r), (m_f, C_f)$는 각각 진짜 데이터의 평균과 분산, 가짜 데이터의 평균과 분산이다(Tr은 대각합). 비슷한 실험으로 동일한 해상도의 이미지를 생성하는 GAN-INT-CLS 모델의 경우 2.88의 Inception score를 달성하였고, 제안된 모델은 3.24 정도의 score를 달성하였다. 두 논문의 차이점은 문장을 임베딩하는 부분에서 큰 차이를 보인다. 즉, 제안된 모델이 문장의 임베딩이 더 잘 되기 때문에 나온 결과라고 할 수 있다.

조선대학교
CHOSUN UNIVERSITY

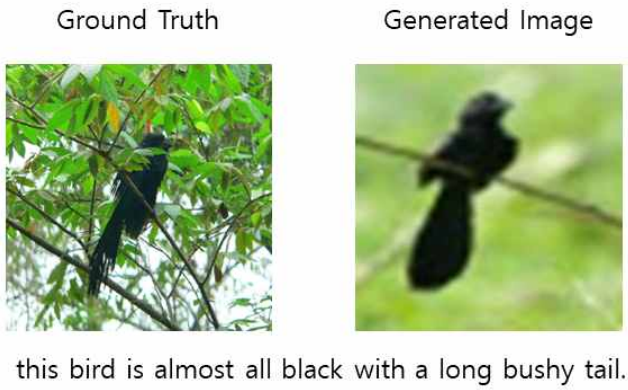

그림 22. Self-Attention GAN의 실험 결과 1

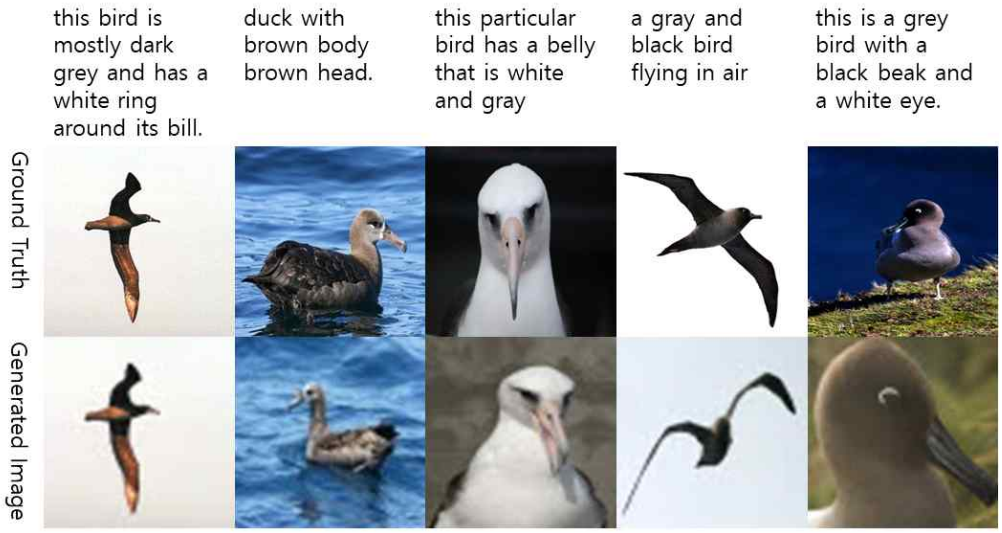

그림 23. Self-Attention GAN의 실험 결과 2

그림 21과 22에 나온 결과를 보면, 해상도가 낮긴 하지만 문장에 있는 각 단어의 특징들을 잘 잡아서 표현하고 있음을 알 수 있다. 그림 22와 23의 결과를 보면 길이가 길수록, 즉 이미지에 대한 설명이 길수록 특징을 잘 잡아 표현하고 있다는 점을 알 수 있다. 그러나 전체적인 이미지에 블러 처리가 된 것처럼 흐릿하다는 문제가 있다. 또한 네트워크가 학습을 할 때 한 가지 문제가 있는데 바로 이미지의 배경이다. 실제 이미지의 특징인 자연에서 찍은 이미지를 사용한

조선대학교
CHOSUN UNIVERSITY

다는 점인데, 새의 모습이 아닌 배경이 학습에 큰 영향을 미치고 있다고 추정된다. 생성된 이미지의 경우 특히 배경을 잘 표현하지 못하며 동시에 대부분의 이미지가 배경이 뭉개진 상태로 표현됨을 알 수 있었다. 또한 하드웨어의 한계로 더 고해상도의 이미지를 표현하는데 한계가 있기 때문에 전체적인 해상도가 떨어짐을 볼 수 있다. 학습에는 약 7일이 소요된다.

그림 24. 설명이 잘 된 경우 생성된 결과

조선대학교
CHOSUN UNIVERSITY

Ⅴ. 결 론

본 논문에서는 딥러닝 기법 중 자연어 처리에 가장 좋은 성능을 보여준 Attention-Mechanism을 사용하여 컴퓨터가 인간의 언어를 이해하고 이에 맞는 이미지를 생성하게 하여 텍스트를 얼마나 문맥에 맞게 이해하였는지 판별하였다. 논문에서 제안하는 Self-Attention GAN은 입력받은 텍스트를 문맥에 맞게 임베딩하는 과정을 거치고, 임베딩 값들을 이용하여 Conditional GAN[8]을 이용하여 이미지로 만들어가는 학습 과정을 거친다. 학습 과정에서 텍스트의 임베딩 값들이 비슷해지는 점을 방지하기 위해 Penalty를 적용하고 이미지를 생성하는 과정에서 기존의 GAN과 다르게 임베딩한 값에 잘못된 이미지를 적용하여 잘못된 이미지와 진짜 이미지를 동시에 학습시켜 문장에 맞는 이미지를 정확하게 생성할 수 있도록 훈련을 수행하였다.

실험의 초기에는 고해상도의 이미지 생성에 초점을 맞추었으나 하드웨어적인 한계 인하여 높은 해상도의 이미지 생성에는 제한점이 있었고 이에 맞추어 낮은 해상도의 이미지 생성에 초점을 맞추어 학습을 진행하였다. 실험 결과 기존의 선행연구와 비교했을 때 Inception Score에서 약 0.36의 성능 향상을 보였다. 이 결과로 미루어 볼 때 선행 연구의 문제점이었던 긴 문장에 취약했던 점이 보완되고 오히려 긴 문장일수록 더 잘 표현되는 모습을 보여주었다. 향후 저해상도 이미지가 생성되는 모델의 취약점을 극복하여 고해상도의 이미지를 표현하는 방법을 만들기 위한 추후 연구가 필요하다고 생각된다.

조선대학교
CHOSUN UNIVERSITY

참고문헌

[1] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[2] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112).

[3] Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

[4] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11), 2673-2681.

[5] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

[6] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

[7] Lin, Z., Feng, M., Santos, C. N. D., Yu, M., Xiang, B., Zhou, B., & Bengio, Y. (2017). A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130.

[8] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. In Advances in neural information processing systems (pp. 2234-2242).

[9] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. (2016). Generative adversarial text to image synthesis. arXiv preprint arXiv:1605.05396.

[10] Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.