유전 알고리즘을 이용한 텍스트 GAN*

오진영 ○ 차정원
창원대학교
psycheojy@cwnu.ac.kr, jcha@cwnu.ac.kr

Genetic Algorithm for Text Generation in GAN Framework

Jinyoung Oh○ Jeong-Won Cha
Changwon National University

요 약

비지도 학습 모델인 GAN은 학습 데이터 구축이 어려운 여러 분야에 활용되고 있으며, 알려진 문제점들을 보완하기 위해 다양한 모델 결합 및 변형으로 발전하고 있다. 하지만 텍스트에 기반한 GAN은 풀어야 할 문제가 많다. 그중 문장 생성 분야에서는 입력 코퍼스 분포를 따르는 토큰(예: 어절, 음절)을 기반으로 생성자가 학습되므로 범위를 벗어난 문장을 생성하려는 경우 목적에 맞지 않는다. 본 논문에서는 유전알고리즘에서 차용한 변이 생성자를 정의하여 입력 코퍼스에 존재하지 않는 다양한 문맥들의 문장 생성 방법을 제안한다.

1. 서 론

생성적 적대 신경망(Generative Adversarial Network, GAN)[1]은 생성자(Generator)와 판별자(Discriminator) 두개의 목적이 충돌하는 신경망을 구성하여 적대적으로 대립하며 학습하는 것을 말한다. 초기 GAN이 발표된 이후 보완 및 변형된 연구가 폭발적으로 늘어난 이유는 비지도 학습을 하는 모델로서 기존에 학습 데이터 구축이 어려운 분야에서 활용될 수 있다는 점이다. 또한, 생성자는 입력 데이터와 유사한 이미지, 음성, 텍스트 등을 생성하기 때문에 생성에 대한 주요 모델로서 여러 도메인에 적용되어 왔다. 텍스트 생성에도 GAN을 이용한 다양한 연구가 있었다. 하지만 연속적인 데이터 값으로 표현이 가능한 이미지에 비해 텍스트는 손실 함수 값을 바탕으로 문장 의미 변화를 주는 학습에 어려움이 있다. [2]는 몬테카를로 탐색과 강화 학습(Reinforcement Learning)으로 정책 그라디언트(policy gradient)를 학습하였고 [3]은 배치 단위의 보상(Reward)으로 학습을 진행하여 그라디언트 포화(saturating)를 보완하였으며, [4]는 판별자의 평가에 순위를 계산하여 이미지에 적용하였다. [5]는 몬테카를로 탐색의 효율을 높이기 위해 판별자가 추출한 정보를 생성자에게 전달(노출) 함으로써 문장 길이와 성능이 높아짐을 실험으로 증명하였다.

대부분의 생성 모델은 입력 데이터를 학습하고 그와 유사한 결과를 생성하기 위한 것이 목적이다. 예를 들어 텍스트 생성에서 입력과 출력 결과물의 문체가 다른 페르소나 문장을 생성하고자 한다면 쌍으로 구성된 학습 문장 기반의 지도 학습이나 외부 데이터를 이용해 모델을 생성하여 활용하는 형태이다. [6]은 함의(Entailment) 문장을 생성하는 것으로 외부 리소스와 수작업으로 구축한 규칙으로 적대적 학습 방법을 함으로써 입력 데이터의 함의 문장을 생성한다. [7]은 부모 생성자에 여러 분포를 가지는 자손 변이 생성자를 만들어 경쟁을 함으로써 높은 평가를 받은 자손 생성자만 남기는 것을 반복하여 모드 붕괴를 막고자 하였지만 새로운 항목을 생성하지 않는다.

본 논문에서는 텍스트 GAN으로 생성자에 변이 생성 로직을 추가하여 입력 데이터에 포함되지 않은 토큰을 생성하고 학습하도록 한다. 또한 (b)와 같이 페르소나를 정의하여 목적에 맞는 문장 생성을 하도록 하였다.

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2018-0-00247, GANs를 이용한 딥러닝용 학습데이터 자가 증식 기술 및 유효성 검증 기술 개발)

(a) 오늘 날씨를 웹에서 검색하였다.
(b) 오늘 날씨를 *인터넷*에서 *찾아봤어요*.

(b)는 (a)에 대한 아이 페르소나 문장이다. '인터넷' '찾아봤어요'는 입력에 포함되지 않은 것이고 학습에도 이러한 단어가 없는 경우 생성 모델에 포함되지 않는데 이를 변이 생성을 통해서 생성하도록 학습하고자 한다.

2. 제안 방법

제안 구조는 그림 1과 같이 생성자(Generator), 판별자(Discriminator)가 있는 GAN의 구조로 생성자는 기본적인 어절 토큰 생성자(G_b)와 변이 생성자(Mutation Generator, G_m)로 구성된다. 생성자는 몬테카를로 탐색을 통하여 문장 후보를 생성한다. 문장 후보를 구성하는 어절 생성 시 먼저 G_b의 결과를 얻고, 유전 알고리즘인 G_m으로 변이를 결정한다. 이 단계마다 CNN 모델인 판별자는 생성된 결과를 판별하여 리워드를 반환하고 탐색 종료에는 강화학습인 정책 그라디언트로 모델에 반영한다. 그림 1에서 붉은색으로 표시된 어절은 G_m 에서 변이로 결정되어 변형이 된 것을 표시한다.

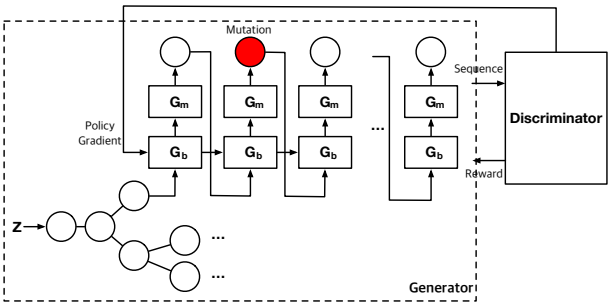

그림 1 유전 알고리즘을 이용한 텍스트 생성

2. 1 유전 알고리즘(Genetic Algorithm)

생성자는 세대에서 한번의 시도로 문장을 생성하는 것이 아니라, 몬테카를로 탐색으로 다양한 문장 후보를 생성한다. 문장 후보 생성 과정에서 각 어절은 G_b와 G_m을 거친 결과물이다. G_b는 노이즈 입력으로 문장을 생성하는 LSTM 모델이며 G_m은 페르소나 정보를 가지고 있는 랜덤 변환기이다. G_m은 G_b의 결과를 입력으로 받아 어절에 변이를 부여할 것인지를 먼저 판단한다. 변이를 부여하고자 하면 어절 내 일부 음절을 변형한다. 변이 부여 여부와 어절 중 변형할 음절 선택은 모두 랜덤으로 정의된다. G_m의 결과가 "먹었다"라면 랜덤의 결정에 의해 변이 여부를 설정하고 음절 '다'를 '어'로 변형하는 것도 랜덤으로 이루어진다. '어'가 아닌 '라', '까', '었' 등의 의미가 통하지 않는 음절을 선택할 가능성이 매우 높다. 하지만 본 연구는 이런 변이를 많이 발생시키고 판별자 리워드로 손실 함수를 반영하여 자연스럽게 네트워크 내에서 도태되거나 살아남을 수 있도록 구성하는 것이다. 변이 중 목적에 맞는 변이를 유지하기 위해서 세대 학습마다 변이형과 페르소나 데이터의 임베딩 정보를 비교하여 유사도가 높은 변이를 다음 학습 단계에서도 유지될 수 있도록 하였다. 또한 변이를 생성하는 음절 치환에서 한글 전체 음절 11,172개가 아닌 페르소나를 구성하는 음절 기반으로 처리함으로써 효율을 높이고자 하였다.

2. 2 생성자 및 판별자 학습

GAN은 생성자와 판별자를 순차적으로 학습한다. 대부분 한 세대(epoch)에서 생성자를 먼저 학습한 뒤, 학습한 생성자의 결과와 실제 코퍼스를 판별하는 것으로 판별자를 학습한다. 본 연구는 판별자 학습 단계에서 생성자(G_b, G_m)의 결과물에도 변이형이 포함될 수 있도록 하였고 변형은 2.1의 처리와 동일하게 진행한다. 즉, 그림 2의 판별자 입력에 붉은색으로 표시한 것과 같이 최종 생성자 결과에 변이가 포함되어 있다. 변이를 추가한 문서로 판별자를 학습함으로써, G_m 에서 동적 구성되는 토큰들의 손실도 계산 할 수 있다. 또한 기존 텍스트 GAN과 다르게 본 논문에서는 생성자를 위한 사전 학습(Pre-training)을 진행하지 않는다. 사전 학습을 함으로써 학습 코퍼스에 포함된 어절과 변이와의 손실의 차이가 나므로 사전 학습을 진행하지 않고 대립 학습을 함으로써 변이 어절이 녹아들 수 있도록 한다.

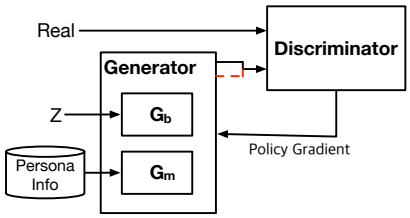

그림 2 구조도

2. 3 손실 함수

손실 함수는 생성자가 생성한 문장의 각 어절 모델 확률, 어절을 생성하기 위해 몬테카를로 탐색을 한 판별자의 리워드 결과로 계산된다. 대부분 손실함수는 이런 경우 타켓 토큰을 원-핫(One-hot) 인코딩하여 1로 부여된 값만 연산하게 된다. 하지만 무시된 나머지 값도 손실 함수에서 유의미한 정보이다. [8]에서는 이런 문제를 고찰하고 방법을 제안하였고, 본 논문에서는 총 어절 n개 중에 정답은 0.9로 설정하고 정답이 아닌 $n-1$개는 0.1을 $n-1$로 나눈 값으로 동일하게 설정한다.

3. 실험

본 논문에서는 유전 알고리즘을 이용한 텍스트 생성 검증을 위하여 입력 코퍼스와 문체가 다른 페르소나 문장 생성 실험을 구성하였다.

3. 1 실험 설정

실험에는 아이 페르소나 코퍼스를 사용한다. 1,855 문장 (원 문장 - 페르소나 문장) 쌍을 구축하였다. 그리고 문장에 포함된 개체명은 레이블링하여 학습

시에는 개체명 어휘를 제외하였다. 또한 2.1에서 정의한 것과 같이 문장 쌍을 네트워크에 바로 학습한 지도학습이 아닌 원 문장만 활용하여 학습하였고, 유전 알고리즘을 위한 G_m 은 페르소나 문장에 포함된 어휘 일부가 사용된다. 페르소나 문장은 최종 평가에서 사용된다. 평가 지표에는 음절 기반 BLEU와 Rouge를 사용하였고, 입력 데이터 전체를 레퍼런스로 사용한다.

3. 2 실험 결과

실험은 2,500번 세대(epoch)을 진행하였다. 먼저 학습 중 얻은 변이 리스트 정보는 표 1과 같다.

표 1 생성된 변이 정보

항목	항목 수	예제
학습 세대	2,500	
총 변이 수	11,842,149 (세대 평균 4,700개)	
중복 제거	5,087,895	
빈도 1,000 이상	306	케떠여, 먹었어, 업떠여, 드렸으, 나봐요, 말이랑, 시락이
빈도 10 이하	5,059,563	아어때요, 있시가, 어실래, 무네모, 어때인, 주랐나

표 2 실험 결과

세대	BLEU-3	BLEU-4	Rouge
1100	0.51851	0.31587	0.31884

평균 4,700개의 변이가 한 세대에 생성되었고, 목적 페르소나와 유사한 음절을 가진 변이가 생성되면 여러 세대에 걸쳐 나타난다(케떠여, 업떠여 등). 낮은 빈도의 변이는 페르소나와 유사도가 낮거나 손실 함수를 통해 제거되었다. 표 2는 2,500번 세대 중 BLEU-4의 성능이 높은 세대의 BLEU, Rouge를 나타내고 그림 3은 전체 세대의 성능 추이를 나타낸다. 음절 기반의 새로운 변이를 추가하기 때문에 음절 기반으로 성능을 측정하였다. 학습 초창기에는 빠르게 성능이 높아지고, 이후에는 유지되며 다양한 변이를 위한 손실 계산을 한다.

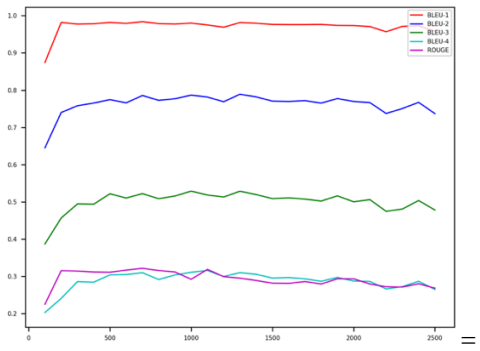

그림 3 성능 추이

마지막으로 그림 4에서는 손실함수에서 토큰의 값을 이진으로 부여한 원-핫 인코딩과 [8]에 기반한 손실 함수 차이를 확인할 수 있다. 손실 함수를 제외한 나머지 데이터 및 매개변수는 동일하다. 가로축은 각각의 변이를 의미하고, 세로축은 손실값을 나타낸다. 각 변이가 같은 값을 많이 부여받은 경우 점의 크기가 커진다. 파란색은 원-핫 인코딩으로 손실 함수를 계산한 것이고 빨간색은 [8]을 의미한다. 파란색은 변이 어절들의 손실 값이 0에 가까운 분포하는 것이 많지만 빨간색은 다양한 값을 가지고 있는 것으로 나타났다. 이것은 변이 중에 의미있는 것은 손실이 반영되어 세대가 바뀌어도 보존될 수 있는 가능성을 높이게 하며 원하는 문장을 생성하는데 효과를 주는 것을 확인 할 수 있다.

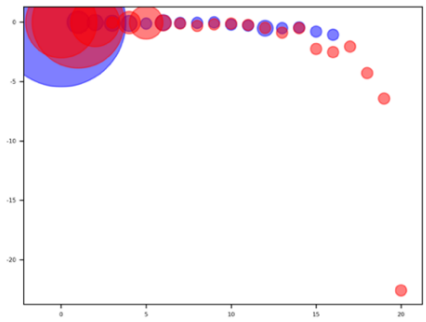

그림 4 손실 함수 비교

4. 결론

본 논문에서는 문장 생성에 유전 알고리즘을 이용하여 새로운 문장을 생성하는 모델을 제안하였다. 본 알고리즘을 페르소나 생성에 적용하여 의미있는 결과를 얻었다. 향후에는 보다 자연스럽고 긴 문장을 생성하기 위해 구조를 보완하려고 한다.

참 고 문 헌

[1] I. Goodfellow et al., “Generative adversarial nets,” In *NIPS*, pp. 2672~2680, 2014.

[2] Lantao Yu et al., “SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient,” in *AAAI*, 2017.

[3] Tong Che et al., “Maximum-Likelihood Augmented Discrete Generative Adversarial Networks,” *arXiv:1702.07983*, 2017.

[4] Rahul Dey et al., “RankGAN: A Maximum Margin Ranking GAN for Generating Faces,” in *ACCV*, 2018.

[5] Jiaxian Guo et al., “Long Text Generation via Adversarial Training with Leaked Information,” in *AAAI*, 2018.

[6] Dongyeop Kang et al., “AdvEntuRe: Adversarial Training for Textual Entailment with Knowledge-Guided Examples,” in ACL, 2018.

[7] Chaoyue Wang et al., “Evolutionary Generative Adversarial networks,” *arXiv:1803.00657*, 2018.

[8] Chang-Uk Shin et al., “범주 불균형 분류 문제를 위한 동적 비용 민감 학습”, in KCC, 2019.