BEGAN 생성 모델의 성능 개선에 관한 연구

박성욱*, 김종찬*, 김도연*
*순천대학교 컴퓨터공학과
e-mail : park7231654@naver.com, dykim@sunchon.ac.kr

A Study on Performance Improvement of BEGAN Generative Model

Sung-Wook Park*, Jong-Chan Kim*, Do-Yeon Kim*
*Dept of Computer Engineering, Sunchon National University

요 약

GAN은 생성된 영상을 실제 영상과 통계적으로 구분 되지 않도록 강제하는 딥러닝 모델이다. BEGAN은 생성자와 판별자의 균형을 유지 및 조절하면서 영상의 잠재 공간을 학습한다. 본 논문에서는 BEGAN의 성능을 개선하기 위해 구조를 변경하고 기존 모델과 성능을 비교했다. 제안한 모델의 성능이 수렴 판정용 함수에서 0.072 수치로 기존 모델보다 높았다.

1. 서론

최근 딥러닝(Deep Learning)[1]에서 회자되는 연구 분야 중 하나로 적대적 생성 신경망(Generative Adversarial Networks, GAN)이 있다[2]. GAN은 생성된 영상을 실제 영상과 통계적으로 구분 되지 않도록 강제하는 딥러닝 모델(Model)이다. BEGAN(Boundary Equilibrium GAN)은 생성자(Generator)와 판별자(Discriminator)의 균형(Equilibrium)을 유지 및 조절하면서 영상의 잠재 공간(Latent Space)을 학습한다[3]. BEGAN은 이전 GAN들과 달리 생성자와 판별자가 훈련을 교대로 실시하지 않는다. 본 논문에서는 BEGAN의 성능을 개선하기 위해 구조를 변경하여 실험해보고, 기존 모델과 성능을 비교했다. 성능 지표로는 수렴 판정용 함수를 사용했다.

2. 관련연구

GAN은 변이형 오토인코더(Variational Auto-Encoder)와는 다른 방법으로 영상의 잠재 공간을 학습한다. GAN의 생성자는 랜덤 벡터(Random Vector)를 입력으로 받아 이를 합성 영상(Synthetic Image)으로 디코딩(Decoding)한다. 판별자 네트워크는 실제 또는 합성 영상을 입력으로 받아 실제 영상인지 합성 영상인지 판별하고, 훈련 피드백(Training Feedback)을 생성자에게 전달한다. Fig 1은 GAN의 기본 훈련 절차를 나타낸다.

생성자에서 생성된 영상의 품질 향상을 위해 DCGAN(Deep Convolutional GAN)이 처음 발표됐다[4]. 이후 판별자를 에너지 함수(Energy Function)로 모델링(Modeling) 하는 EBGAN(Energy Based GAN)이 발표됐다[5]. EBGAN은 수렴이 안정적이며 하이퍼파라미터(Hyper-parameter) 변형에 강건하다. EBGAN도 BEGAN과 같이 판별자가 오토인코더(Auto-Encoder, AE)다. 최근에 발표된 WGAN(Wasserstein GAN)은 모델의 수렴 척도로 이용할 수 있는 손실 함수(Loss Function)를 도입했다[6]. WGAN은 훈련 속도는 느리지만 안정적이다. PGAN(Progressive Growing of GAN)은 고품질 영상을 생성하는 또 다른 모델이다[7]. PGAN은 [1]의 손실 함수를 수정하는 대신 새로운 훈련 절차로 학습의 안정성을 높였다.

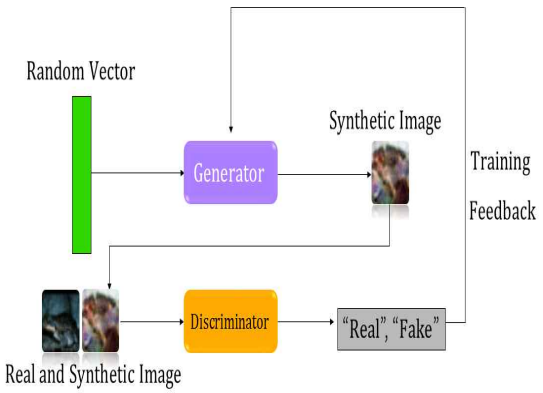


Fig 1. The training process of a GAN.

3. 개선된 BEGAN의 구조와 원리

판별자로 AE를 사용하고 데이터 분포가 아닌 손실 분포를 일치시킨다는 점은 변경하지 않았다. 또한, 생성자와 판별자가 균형을 맞추기 위해 훈련을 진행한다. BEGAN은 근본적으로 WGAN의 훈련 목적과 유사하다. 개선된 BEGAN 구조는 Fig 2와 같다.

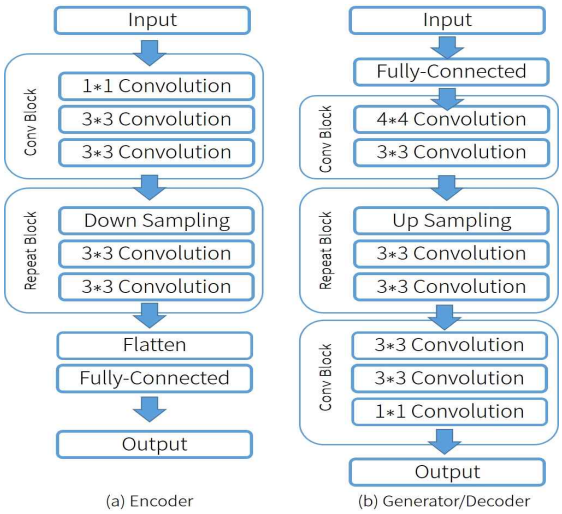


Fig 2. Improved BEGAN structure.

1~4 컨벌루션(Convolution) 연산과 활성 함수(Activation Function)로 Leaky ReLU(Rectified Linear Unit, LReLU)를 사용했다[8]. LReLU의 알파(Alpha)값은 0.2로 설정했다. 반복 블록(Repeat Block) 수의 경우 인코더(Encoder)는 3번, 생성자 및 디코더(Decoder)는 2번으로 설정했다. 컨벌루션 층의 필터(Filter) 개수는 각 다운샘플링(Down-sampling)마다 선형적으로 증대된다. 다운샘플링은 스트라이드(Stride) 2로, 업샘플링(Up-sampling)은 k-최근접 이웃(k-Nearest Neighbors) 알고리즘에 의해 수행된다. 인코더와 디코더 경계에서 처리된 데이터는 완전 연결(Fully-connected) 층을 통해 매핑(Mapping)된다. 입력 벡터(Vector) z는 [-1,1] 사이의 균등 분포(Uniform Distribution)에서 랜덤(Random)하게 추출된다. 전체 구성도는 Fig 3과 같다.

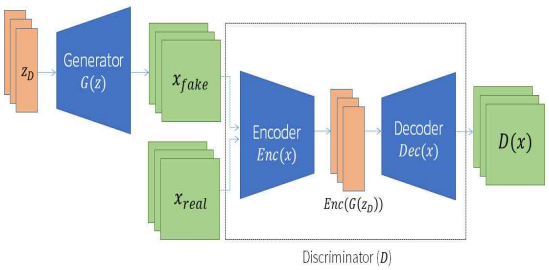


Fig 3. An overview of BEGAN.

4. 실험 및 성능평가

4.1 H/W 및 S/W 사양

소프트웨어 사양의 경우, 운영체제(Operating System)는 우분투(Ubuntu) 16.04.4 LTS, CUDA(Compute Unified Device Architecture)는 9.0.176, cuDNN(cuda Deep Neural Network library)은 7.1, 텐서플로(Tensorflow)는 1.12.0, 케라스(Keras)는 2.2.4, 파이썬(Python)은 3.5.2를 이용했다. 케라스는 텐서플로 외에도 다양한 딥러닝 라이브러리(Library)를 호출하여 그에 맞는 알고리즘을 수행한다. 텐서플로는 머신 러닝(Machine Learning)과 딥러닝을 위한 오픈소스(Open Source) 라이브러리다. 실험에 이용된 하드웨어 사양은 Table 1과 같다.

Table 1. Hardware specifications.

	Specifications
CPU	Intel Core i7 7700K
Graphics Card	NVIDIA TITAN Xp 12GB
RAM	Samsung DDR4 32GB
SSD	Samsung 850 Pro 512GB

4.2 실험 환경

옵티마이저(Optimizer)의 경우, 적응적 모멘트 추정(Adaptive moment estimation, Adam)을 이용했고[9], 학습률(Learning Rate)='0.001', 일차 모멘텀(Momentum)용 계수 베타(Beta)1='0.9', 이차 모멘텀용 계수 베타2='0.999', 엡실론(Epsilon)="None", 가중치 감쇠(Weight Decay)='0.0', AMSGrad="False"로 설정했다. 가중치 초기화(Weight Initialization)의 경우, 글로르트 균등 분포(Glorot Uniform Distribution) 방법을 이용했다. 생성자와 판별자의 학습률은 동일하다.

입력 벡터 z의 크기는 32, 감마(Gamma)는 0.5로 설정했다. 감마는 다상비(Diversity Ratio)를 결정하는 변수로 사용한다. $L(x)$와 $L(G(z_D))$를 어느 정도 중요하게 다룰지 나타내는 가중치 k_t는 0.0, k_t의 학습률 λ_k는 0.001로 설정했다. x는 실제 영상, $L(x)$는 실제 영상에 대한 오차(Error)다. z_D는 판별자가 학습할 때 생성자에 입력할 랜덤 벡터, $L(G(z_D))$는 생성자에 의해 생성된 영상의 오차다. $L(x)$와 $L(G(z_D))$의 출력 과정은 Fig 4와 같다.

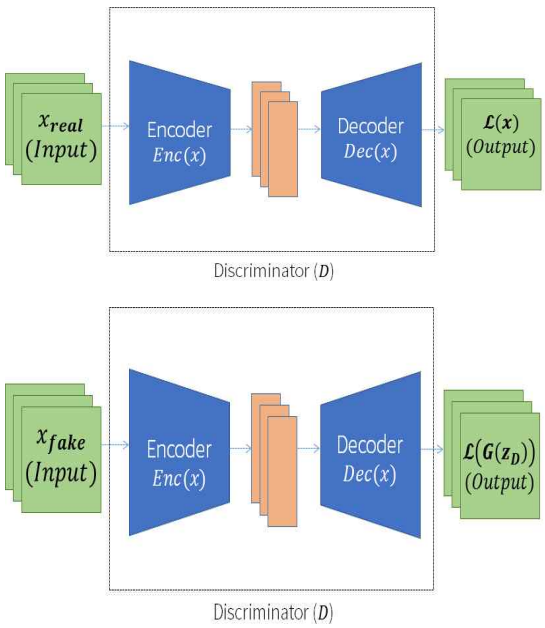


Fig 4. Two types of discriminator inputs and outputs.

글로벌 스텝(Global Step)은 220,000, 데이터 세트는 360 K Celebrity Face가 아닌 CelebA(large-scale Celeb faces Attributes)를 이용했다. CelebA는 202,599장의 유명 인사 얼굴 영상을 포함하고 있는 벤치마크(Benchmark) 데이터 세트다. 미니 배치(Mini Batch) 크기는 128, 훈련 및 예측 영상의 해상도는 64×64 픽셀(Pixel)로 설정했다. 본 논문에서는 위와 같은 설정을, 도출한 결론의 객관성 확보를 위해 두 모델에 동일하게 적용했다. 실험에 이용된 두 모델의 파라미터(Parameter) 수는 Table 2와 같다.

Table 2. Number of parameters for BEGAN and Ours model.

Model	Generator	Discriminator	
		Encoder	Decoder
BEGAN	432,323	2,960,608	432,323
Ours	864,963	2,680,608	864,963

생성자의 파라미터 수는 변경된 모델이 864,963개로 변경 전 모델보다 432,640개 더 많고, 판별자의 인코더 파라미터 수는 변경된 모델이 2,680,608개로 변경 전 모델보다 280,000개 적다. 전체 파라미터 수는 변경된 모델이 변경 전 모델보다 585,280개 더 많다.

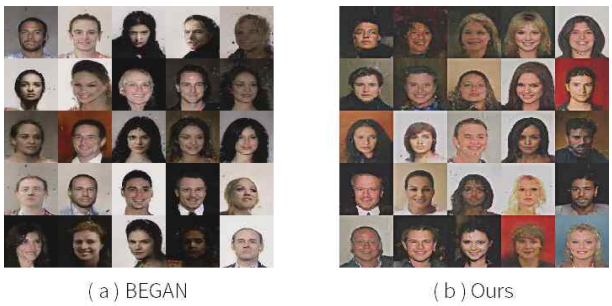
4.3 실험 결과

BEGAN은 식 (1)을 이용하여 모델이 수렴했는지 아닌지 판단한다.

$$M_{global} = L(x) + \left|\gamma L(x) - L(G(z_G))\right| \qquad (1)$$

M_{global}은 글로벌 측정(global Measurement, M), z_G는 생성자가 학습할 때 생성자에 입력할 랜덤 벡터를 의미한다. 식 (1)은 실제 영상의 오차 $L(x)$가 작은 것과 $\gamma L(x) - L(G(z_G))$가 0에 가까운 것으로 판정할 수 있다. 간단히 말해서, M이 0에 가까우면 수렴한다고 판정할 수 있다.

훈련 시간은 두 모델 모두 약 이틀 정도 소요됐다. Fig 5는 결과 영상을 나타난다. 제안한 모델의 성능은 수렴 판정용 함수 M에서 0.072 수치로 기존 모델보다 높았다.



(a) BEGAN (b) Ours

Fig 5. Experimental results(64×64 with 64 filters).

5. 결론 및 향후 과제

본 논문에서는 기존 BEGAN의 생성자와 인코더 및 디코더 구조를 변경하고 기존 모델과 성능을 비교했다. 실험 결과, 제안한 모델의 성능이 수렴 판정용 함수에서 0.072 수치로 기존 모델보다 높았다. 모델에 따라 최적의 하이퍼 파라미터는 달라질 수 있기 때문에 추가 훈련 및 하이퍼 파라미터 값 조정을 통해 개선 모델은 더 높은 성능을 얻을 수 있을 것으로 기대된다. 또한 본 논문에서는 배치 정규화(Batch Normalization), 드롭아웃(Dropout), 트랜스포즈 컨벌루션(Transpose Convolution), 컨벌루션 층의 필터 개수 증대와 같은 방법을 적용하지 않았지만, 해당 요소들이 성능을 더 향상시킬 수 있다고 판단된다.

향후 연구 과제로 최적의 잠재 공간 크기와 입력 영상에 노이즈(Noise)를 추가해야 되는 이상적인 시점에 대해 명확하게 밝힐 수 있는 연구가 필요하다.

참고문헌

[1] L. Yann, B. Yoshua, and H. Geoffrey, “Deep Leaning,” Nature, Vol. 521, No. 7553, pp. 436-444, 2015.

[2] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, et al., “Generative adversarial nets,” NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems, Vol. 2, pp. 2672-2680, 2014.

[3] D. Berthelot, T. Schumm, and L. Metz, “BEGAN: Boundary Equilibrium Generative Adversarial Networks,” arXiv, arXiv:1703.10717, 2017.

[4] A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” arXiv, arXiv:1511.06434, 2015.

[5] J. Zhao, M. Mathieu, and Y. LeCun, “Energy-based Generative Adversarial Network,” arXiv, arXiv:1609.03126, 2017.

[6] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein GAN,” arXiv, arXiv:1701.07875, 2017.

[7] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation,” arXiv, arXiv:1710.10196, 2017.

[8] A.L. Maas, A.Y. Hannun, and A.Y. Ng, “Rectifier Nonlinearities Improve Neural Network Acoustic Models,” Proceeding of International Conference on Machine Learning, 2013.

[9] D.P. Kingma and L.J. Ba, “Adam: A Method for Stochastic Optimization,” Proceeding of International Conference on Learning Representations, arXiv:1412.6980, 2015.