J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr
Copyright ⓒ 2020 KSII

군 폐쇄망 환경에서의 모의 네트워크 데이터 셋 평가 방법 연구

A study on evaluation method of NIDS datasets in closed military network

박 용 빈[1*] 신 성 욱[1] 이 인 섭[1]
Yong-bin Park Sung-uk Shin In-sup Lee

요 약

이 논문은 Generative Adversarial Network (GAN) 을 이용하여 증진된 이미지 데이터를 평가방식인 Inception Score (IS) 와 Frechet Inception Distance (FID) 계산시 inceptionV3 모델을 활용 하는 방식을 응용하여, 군 폐쇄망 네트워크 데이터를 이미지 형태로 평가하는 방법을 제안한다. 기존 존재하는 이미지 분류 모델들에 레이어를 추가하여 IncetptionV3 모델을 대체하고, 네트워크 데이터를 이미지로 변환 및 학습 하는 방법에 변화를 주어 다양한 시뮬레이션을 진행하였다. 실험 결과, atan을 이용해 8 * 8 이미지로 변환한 데이터에 대해 1개의 덴스 레이어 (Dense Layer)를 추가한 Densenet121를 학습시킨 모델이 네트워크 데이터셋 평가 모델로서 가장 적합하다는 결과를 도출하였다.

☞ 주제어 : 데이터 셋, 네트워크 침입 탐지 시스템, 데이터 평가, 머신러닝

ABSTRACT

This paper suggests evaluating the military closed network data as an image which is generated by Generative Adversarial Network (GAN), applying an image evaluation method such as the InceptionV3 model-based Inception Score (IS) and Frechet Inception Distance (FID). We employed the famous image classification models instead of the InceptionV3, added layers to those models, and converted the network data to an image in diverse ways. Experimental results show that the Densenet121 model with one added Dense Layer achieves the best performance in data converted using the arctangent algorithm and 8 * 8 size of the image.

☞ keyword : dataset, Network Intrusion Detection System, Data evaluation, Machine Learning

1. 서 론

Generative Adversarial Network (GAN) 과 이에 파생된 다양한 종류의 GAN 연구가 많이 진행되면서 이미지를 증진하고, 증진된 이미지를 평가하는 방법 역시 요구되고 있다. 기존에 존재하는 방법은 Imagenet의 이미지를 학습시킨 InceptionV3 모델을 기반으로 한 Inception Score (IS) 와 이를 보강하여 만들어진 Frechet Inception Distance (FID) 가 대표적이다.

GAN은 이미지 외에도 여러 형태의 데이터를 증진하는 데에도 사용된다. 군에서는 군 환경에 맞는 Network Intrusion Detection System (NIDS)를 개발하고 성능을 평가하기 위해 NIDS 데이터 셋이 필요하다. 그러나 군 기밀 및 보안 특성 상 실제 네트워크 데이터를 통한 평가는 많은 제약이 따른다. 따라서 GAN을 통해 군 환경과 유사한 NIDS 데이터를 증진하여 이를 대체한다. 그리고 증진된 NIDS 데이터에 대해서 실제 데이터와 유사한 지 평가해야 한다.

악성코드 분야에서는 악성코드를 grayscale 이미지로 변환한 후에 GAN 으로 학습시켜 악성코드를 증진하는 연구나, 이미지 분류 모델을 이용하여 분류하는 연구가 많이 진행되었다. 머신러닝에 사용할 대상을 이미지로 변환해 이미지 분류 모델을 이용한다는 점을 착안하여, NIDS 데이터 셋을 이미지로 변환 후, 기존 InceptionV3 모델에 학습시켜 IS 나 FID를 통해 평가를 시도 해보았다. 그러나 일반 사물 사진 이미지와 달리 이미지 내의 인접하는 픽셀 값의 연관성이 작아 InceptionV3 모델의 IS, FID 로 의미 있는 값을 도출하기 어렵다. 또한 이미지의 픽셀 값은 $[0,255]$의 정수 값을 가져야 하는데 NIDS 데이터 셋 값의 범위는 $(-\infty,\infty)$이다. 그러므로 NIDS 데이터를 어떻게 $[0,255]$의 정수로 치환하고 이미지로 변환하느냐에 따라 정보량이 달라지고 이는 이미지 분류 모델 학습율에 영향을 미치게 된다.

1 2nd R&D Institute, ADD, Seoul, 05661, Korea.
* Corresponding author (sinen0308@add.re.kr)
[Received 14 November 2019, Reviewed 21 November 2019(R2 2 January 2020), Accepted 29 January 2020]

http://dx.doi.org/10.7472/jksii.2020.21.2.121

따라서 이번 연구에서는 NIDS 데이터를 정보의 손실을 최소화 하여 이미지를 변환하는 방법과 InceptionV3 모델 대신에 기존에 존재하는 이미지 분류 모델에 변화를 주어 학습을 진행한다. 각각 학습된 모델에 대해서 증폭된 이미지의 IS, FID를 구하여 최적의 평가 방법을 모색한다.

2. 관련 내용

2.1 Generative Adversarial Network(GAN)

GAN은 2014년 NIPS에서 Lan Goodfellow 가 발표한 회귀생성 모델로서 분류를 담당하는 모델 (판별자 D) 과 회귀 생엉을 담당하는 두 개의 모델 (생성자 G) 로 구성이 되어있다. 생성자와 판별자가 서로의 성능을 개선해 적대적으로 경쟁해 나가는 모델이다.

흔하게 경찰과 지폐 위조범의 대립으로 비유할 수 있다. 지폐 위조범의 목표는 위조 지폐를 만들어 경찰을 속이는 것이고, 경찰은 지폐가 가짜인 지 진짜인 지 구별을 하게 된다. 이러한 경쟁이 지속적으로 학습되면 결과적으로는 진짜와 위조 지폐를 구별할 수 없는 정도의 상태가 되며, 진짜와 거의 차이가 없는 위조지폐를 생성할 수 있게 된다. 수학적으로 생성자는 앞에서 말한 원 데이터의 확률 분포를 알아내려고 노력하며, 학습이 종료된 후에는 원 데이터의 확률 분포를 따르는 새로운 데이터를 만들어낸다.

2.2 Malware image visualization and classification

악성코드 분석에는 정적 분석, 동적 분석 등 다양한 분석이 있다. 악성코드는 8bit 단위로 grayscale 이미지로 변환할 경우 바이너리의 섹션마다 차이가 쉽게 보인다. 유사한 속성을 가진 악성코드 일수록 변환된 이미지가 비슷하다. 따라서 쉽게 그룹을 이루거나 비교할 수 있다.

Nataraj, L. (2011)의 연구에서는 이미지들을 k-nearest neighbors 나 Euclidean distance를 통해 분류를 시도했다. k=3 인 knn-classifier 인 경우, 분류의 정확도는 약 98%를 보였다 [6]. S. Choi (2017)의 연구에서는 악성코드를 256 * 256 이미지로 변환하여 CNN를 학습시켰다. 10000개의 정상파일과 2000개의 악성코드 이미지 대상으로 분류를 진행했을 때, 약 95% 이상의 정확도를 보였다 [9].

2.3 NIDS 데이터 셋

2.3.1 KDD99

KDD99은 제 3차 'International Knowledge Discovery and Data Minings Tools Competition'에서 네트워크 침입을 탐지하고 공격과 정상 연결을 구분할 수 있는 예측 모델을 구축하는 목표로 사용된 데이터 셋이다. 군사 네트워크 환경에서 시뮬레이션 된 다양한 공격에 해당되는 데이터가 포함되어져 있으며, IDS 및 머신 러닝 분야에서 가장 많이 사용되었다.

2.3.2 NSL-KDD

이 연구에서는 KDD99 보다 NSL-KDD 가 학습 시간과 분석 면에 유리하다고 판단하여 NSL-KDD를 사용하였다. NSL-KDD는 KDD99의 데이터의 중복성을 제거하고 데이터 셋의 통계적인 부분과 더 유효하고 중요한 공격일수록 많이 학습할 수 있게 개선하였다. 이 데이터 셋은 30개의 넘는 공격이 라벨링이 되어 있지만 이 중에서 군 폐쇄망에 맞는 유형의 공격 라벨링을 28개로 축소하여 학습을 진행하였다.

2.4 기존 평가 지표

2.4.1 Inception Score (IS)

IS는 GAN 으로부터 생성된 이미지의 품질을 평가하기 위한 Google에서 개발한 지표이며, Google의 InceptionV3 모델에 증폭된 이미지를 예상하여 나오는 확률 벡터를 바탕으로 평가하는 방식이다.

IS를 계산하기 위해서 KL divergence를 이용하며 이는 확률 분포가 얼마나 유사하거나 다른 지를 의미한다. IS는 높은 값을 가질수록 좋은 품질이다. InceptionV3가 아닌 다른 이미지 분류 모델에서 확률 벡터를 얻을 수 있다면 수식 (1)을 통해 계산할 수 있다.

$$IS(G) = \exp(E_x KL(p(y|x) \| p(y)))\text{----------}(1)$$

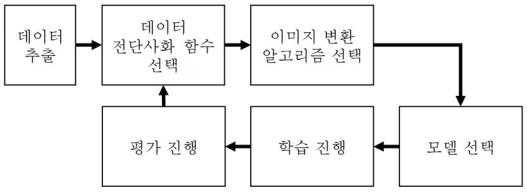
2.4.2 Frechet Inception Distance (FID)

FID도 IS 와 마찬가지로 생성된 이미지의 품질을 평가하고 GAN 의 성능을 측정하기 위한 지표이다. FID는 IS가 실제 이미지와 비교를 하지 않는 점을 보완하기 위해 착안되었다.

$$FID = \|\mu_1 - \mu_2\|^2 + Tr(C_1 + C_2 - 2*\sqrt{C_1 C_2}) \text{----(2)}$$

C는 실제와 가상의 이미지 벡터들의 공분산 행렬을 의미한다. FID은 값이 낮을 수록 좋은 품질을 의미한다. IS와 마찬가지로 InceptionV3가 아닌 다른 이미지 분류 모델에서 실제 데이터 이미지의 확률 벡터와 증폭된 이미지의 확률 벡터를 얻을 수 있다면 수식 (2)를 통해 계산할 수 있다.

3. 연구 결과



(그림 1) 연구 순서
(Figure 1) Order of research

3.1 연구 순서

3.1.1 데이터 수치화

NSL-KDD feature 값 중 'protocol_type', 'services', 'flag', 'label' 의 데이터 값은 숫자가 아닌 문자열로 표현된다. 이미지로 변환하기 위해 문자열 형식의 데이터 값을 실수 형식의 데이터 값으로 변환해야 한다. 그러므로 각 feature 별로 문자열 값을 0부터 정수 값으로 할당한 후, 일대일 대응으로 치환한다.

3.1.2 수치화된 데이터 전단사화

이미지의 픽셀 값은 [0, 255] 범위를 가지고 있으나, 수치형 데이터는 $(-\infty, \infty)$범위에 분포해있다. 이미지로 변환하기 위해서 3.1.1을 거친 수치형 데이터를 함수를 통해 $(-\infty, \infty) \rightarrow [0, 255]$로 매핑한다. $(-\infty, \infty)$에서 유한한 범위로 축소시킬 수 있는 대표적인 함수는 수식 (3), 수식 (4)가 있다.

$$y = \tanh(\theta) \text{ ----------(3)}$$
$$y = atan(\theta) \text{ ----------(4)}$$

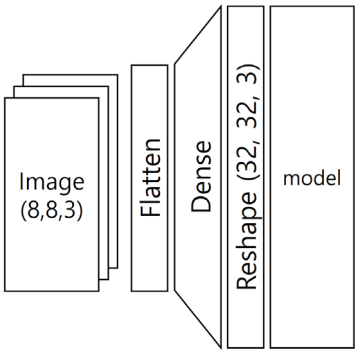
이 두 함수를 수식 (5), 수식 (6) 와 같이 변형하여 $(-\infty, \infty) \rightarrow (0, 255)$ 매핑을 진행했다.

$$y = \tanh(x/255)*255 \text{ ----------(5)}$$
$$y = atan(x*\pi/510)*(510/\pi) \text{------(6)}$$

3.1.3 학습 이미지로 변환

3.1.2 과정을 거친 값을 grayscale 이미지로 변환 하는 과정에 총 두 가지의 처리가 필요하다. 첫 번째는 수식 (5), 수식 (6)을 거친 값은 소수이므로 이미지 변환을 위해 이러한 값들을 정수로 변환해야 한다. 소수 점 이하의 수는 정수 값에 따라 비중이 다르기 때문에 단순한 올림이나 내림, 반올림은 많은 정보량의 손실을 발생한다. 이 문제를 해결하기 위해 소수의 정수 부분과, 소수 점 이하의 수에 255를 곱한 정수 부분, 즉 $[0, 255)$ 값을 갖는 두 정수로 표현해 최대한 정보량의 손실을 줄이고자 시도하였다.

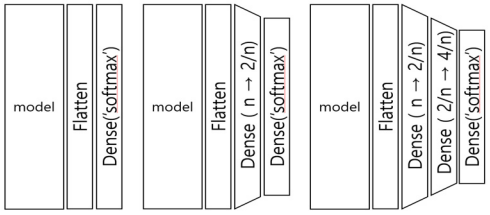
두 번째는 머신 러닝이 학습할 수 있는 이미지의 크기로 변환해야 한다. 기존 이미지 분류 모델의 최소로 요구하는 입력 이미지 크기는 32 * 32이다. 첫 번째 과정을 거치면 41개의 feature 값과 소수 처리로 22개의 정수 값이 더 생겨나므로 총 픽셀로 변환할 수 있는 값은 63개이다. 그 중 하나의 값을 255 패딩 값을 넣어 총 64개로 맞추어 8 * 8 의 이미지로 변환한다. 이 이미지의 픽셀 하나 당 동일한 값의 4 * 4 로 확장해서 변환하면 32 * 32 로 입력 사이즈 크기를 동일하게 만들 수 있다. 혹은 덴스 레이어를 각 모델 앞에 추가하여 8 * 8 의 이미지를 입력받게 할 수 있다. 이 두 가지 방법을 모두 사용하여 평가하고 비교해 보았다.



(그림 2) 모델 구성
(Figure 2) Architecture of model

3.1.4 모델 커스터마이징 방법 결정

이번 연구에 사용될 이미지 분류 모델은 DenseNet121, VGG19, NASNetMobile, MobileNet, ResNet50 이다. 이 각각의 모델 자체는 분류 기능이 없고 마지막 레이어의 결과 값의 출력한다. 따라서 덴스 레이어를 추가해 28개의 라벨에 대해서 확률 벡터를 생성한다. 덴스 레이어를 이용하여 출력 사이즈를 줄이는 정도에 따라 정확도가 달라지기 때문에 덴스 레이어의 개수를 최대 2개까지 더 증가시키며 IS와 FID 변화를 관찰했다.



(그림 3) 모델 커스터마이징 3가지 방법
(Figure 3) 3 ways of model customizing

하나의 이미지 분류 모델 당 총 3가지의 모델로 커스터마이징을 진행했으며 첫 번째는 플랫튼 레이어를 통해 바로 확률 벡터를 생성하고, 두 번째는 덴스 레이어를 하나 더 추가해 출력 수가 입력 수의 절반이 되도록, 세 번째는 출력 수가 입력의 수의 1/2로 만드는 2개의 덴스 레이어를 추가하여 출력 수가 입력 수의 1/4배가 되도록 하였다.

(표 1) 학습 셋 라벨 별 데이터 개수
(Table 1) number of data by training set label

label	#	label	#
normal	67,343	apache2	516
back	956	bof	30
ftp_write	8	guess_pw	840
imap	11	ipsweep	3,599
land	18	loadmodule	9
mscan	697	multihop	7
nmap	1,493	perl	3
phf	4	pod	201
portsweep	2,931	processtable	480
ps	10	rootkit	10
saint	223	satan	3,633
snmp getattack	124	warez master	635
teardrop	892	warezclient	624
snmpguess	232	etc	40

3.1.5 데이터 학습

28개의 라벨의 데이터에 대하여 3.1.4 이후로 변환된 학습 셋 이미지의 개수는 (표 1)과 같다. 변환된 이미지를 3.1.4에서 커스터마이징 된 15 (5 * 3) 개의 모델마다 최대 1000 epoch 까지 학습 셋을 학습을 시키고 100 epoch 마다 모델을 저장을 하였다.

3.1.6 FID, Inception Score, Accuracy 평가

평가를 위한 이미지를 증진하기 위해 wasserstein GAN (WGAN) 으로 1000 epoch까지. 100 epoch 마다 총 10회 이미지를 생성하였다. 1회 이미지 생성 시, 28개의 라벨 별로 강 500장씩 14,000장의 이미지, 총 140.000장의 이미지를 생성하였다. 그리고 각 epoch 별로 14,000장의 시험 셋 이미지로 IS, FID를 측정하였다.

3.2 연구 결과

3.2.1 accuracy를 기반으로 기준 모델 선정

3.1 연구 순서에 따라 3.1.2 전단사화 함수, 3.1.3 이미지 크기 변환, 3.1.4 커스터마이징 모델, 3.1.5 데이터 학습 3.1.6 의 모든 epoch마다 증진된 이미지에 대하여 총 60번의 1000 epoch 학습과 6000 (2*2*5*3*10*10) 번의 FID, IS 평가가 필요하다. 이 모든 평가와 학습을 진행하기에는 시간적, 물리적 자원이 부족하다.

(표 2) atan함수 이용하고 32 * 32로 확장된 이미지를 사용하여 DenseNet 각 모델의 epoch 별 정확도
(Table 2) Accuracy per epoch of each DenseNet models trained by images which are expanded to 32 * 32 using atan function.

model	accuracy (%) by epoch				
DenseNet121	100	200	300	400	500
	76.39	76.72	77.36	76.33	79.54
	600	700	800	900	1000
	78.31	78.75	76.75	79.19	79.34
DenseNet121 + 1 Dense layer	100	200	300	400	500
	74.49	78.20	78.90	78.67	79.72
	600	700	800	900	1000
	79.21	80.22	78.10	78.46	78.10
DenseNet121 + 2 Dense layer	100	200	300	400	500
	75.24	76.19	77.99	76.94	77.61
	600	700	800	900	1000
	77.25	77.40	78.58	78.59	77.23

(표 3) atan함수 이용하고 32 * 32로 확장된 이미지를 사용하여 ResNet50 각 모델의 epoch 별 정확도
(Table 3) Accuracy per epoch of each ResNet50 models trained by images which are expanded to 32 * 32 using atan function.

model	accuracy (%) by epoch				
ResNet50	100	200	300	400	500
	76.04	75.52	75.06	76.66	75.93
	600	700	800	900	1000
	77.04	76.82	77.44	77.12	75.73
ResNet50 + 1 Dense layer	100	200	300	400	500
	75.97	75.61	75.82	78.11	75.90
	600	700	800	900	1000
	75.61	76.20	76.33	74.62	57.59
ResNet50 + 2 Dense layer	100	200	300	400	500
	75.48	76.94	76.79	75.15	76.38
	600	700	800	900	1000
	75.27	74.59	76.91	75.68	73.95

(표 4) atan함수 이용하고 32 * 32로 확장된 이미지를 사용하여 NASNet 각 모델의 epoch 별 정확도
(Table 4) Accuracy per epoch of each NASNet models trained by images which are expanded to 32 * 32 using atan function.

model	accuracy (%) by epoch				
NASNetMobile	100	200	300	400	500
	77.53	72.37	79.20	78.07	75.95
	600	700	800	900	1000
	78.53	78.22	79.06	77.64	77.47
NASNetMobile + 1 Dense layer	100	200	300	400	500
	42.18	74.16	78.90	75.99	76.20
	600	700	800	900	1000
	78.23	77.30	79.87	79.47	78.74
NASNetMobile + 2 Dense layer	100	200	300	400	500
	43.42	71.98	74.92	76.27	76.57
	600	700	800	900	1000
	76.85	77.03	77.16	79.91	78.19

그러므로 5개의 이미지 분류 모델마다 각각 3개씩 덴스 레이어로 커스텀을 시킨 총 15개의 모델에 atan 전단사화 함수를 사용하고 32 * 32 로 생성된 이미지를 1000epoch 까지 100epoch 단위로 데이터 학습을 시킨 후 저장한다. 그리고 각 15개의 학습된 모델을 100 epoch 마다 이미지로 변환된 NSL-KDD 시험 셋을 분류하는 정확도를 계산한 후, 이미지 분류 모델 별로 가장 정확도가 높은 데이터 학습의 epoch 와 덴스 레이어 개수를 기준으로 선택한다. 다음으로 기준에서 전단사화 함수 혹은 이미지 변환하는 방법에 변화를 주어 FID, IS를 평가하고 대조한다. 그 결과, 최대 25번의 1000 epoch 학습과 150번의 FID, IS 평가 가 필요하다. 3.2.2 과 3.2.3을 통해 atan 과 tanh를 전단사화 함수로 사용했을 때의 평가 성능 비교가 가능하며 3.2.3과 3.2.4를 통해 32 * 32 크기의 학습 이미지로 변환했을 때와 8 * 8 크기의 학습 이미지로 변환 했을 때의 평가 성능을 비교할 수 있다.

(표 5) atan함수 이용하고 32 * 32로 확장된 이미지를 사용하여 VGG19 각 모델의 epoch 별 정확도
(Table 5) Accuracy per epoch of each VGG19 models trained by images which are expanded to 32 * 32 using atan function.

model	accuracy (%) by epoch				
VGG19	100 - 1000				
	65.88				
VGG19 + 1 Dense layer	100 - 1000				
	1.07				
VGG19 + 2 Dense layer	100	200	300	400	500
	77.86	77.07	77.30	76.05	76.25
	600	700	800	900	1000
	75.55	76.31	75.98	77.58	74.87

(표 6) atan 함수 이용하고 32 * 32로 확장된 이미지를 사용하여 MobileNet 각 모델의 epoch 별 정확도
(Table 6) Accuracy per epoch of each MobileNet models trained by images which are expanded to 32 * 32 using atan function.

model	accuracy (%) by epoch				
MobileNet	100	200	300	400	500
	78.76	78.86	77.82	77.69	76.58
	600	700	800	900	1000
	78.28	77.65	76.70	79.18	78.64
MobileNet + 1 Dense layer	100	200	300	400	500
	78.29	75.71	76.40	77.31	76.84
	600	700	800	900	1000
	76.58	75.91	77.46	75.52	77.52
MobileNet + 2 Dense layer	100	200	300	400	500
	77.63	76.32	76.75	77.13	76.75
	600	700	800	900	1000
	75.99	77.17	77.72	77.61	76.67

(표 2), (표 3), (표 4), (표 5), (표 6)는 각각의 커스텀된 모델에서 epoch 별 학습 정확도 측정 결과이다. 시험 셋

에서 normal label 데이터 량이 전체에서 65.88%를 차지하고 있기 때문에 정확도는 65.88% 이상이 되어야 최소 한의 분류 능력을 수행한다고 판단할 수 있다. 각 표에서 노란색으로 표시한 부분은 각 이미지 분류 모델 중에 가장 정확도가 높은 값이다. 각 모델마다 가장 높은 정확도를 가진 덴스 레이어의 수와 학습 epoch 수는 모두 다름을 보인다.

DenseNet121 는 덴스 레이어를 한 개를 추가로 붙이고 700 epoch 학습한 상태에서 정확도가 80.22%로 가장 높은 정확도를 보였다. VGG19는 덴스 레이어를 두 개를 추가로 붙이고 100 epoch 학습한 상태에서 77.86% 로 가장 높은 정확도를 보였다. 한 개의 덴스 레이어를 붙였을 때는 모든 epoch에서의 정확도가 1.07%로 최소한의 분류 능력을 가지지 못했다. MobileNet은 덴스 레이어를 추가 없이 900 epoch 학습한 상태에서 79.18%로 가장 높은 정확도를 보였다. NASNetMobile 은 덴스 레이어를 두 개를 추가로 붙이고 900 epoch 학습한 상태에서 79.91% 로 가장 높은 정확도를 보였다. ResNet50은 덴스 레이어를 한 개를 추가로 붙이고 400 epoch 학습한 상태에서 정확도가 78.11%로 가장 높은 정확도를 보였다. 그리고 각 이미지 분류 모델 별로 가장 높은 정확도를 보인 상태에 대하여 기준을 설정하였다.

(표 7) atan함수 이용하고 학습에 32 * 32 확장한 이미지를 사용한 모델들의 IS 값

(Table 7) IS per epoch of each optimized 5 models trained by images which are expanded to 32 * 32 using atan function.

model	IS by epoch				
DenseNet121 + 1 Dense layer + 700 epoch	100	200	300	400	500
	1.20	1.96	2.23	2.41	2.56
	600	700	800	900	1000
	2.71	2.74	2.79	3.12	2.87
VGG19 + 2 Dense layer + 100 epoch	100	200	300	400	500
	1.49	3.83	4.25	4.37	4.55
	600	700	800	900	1000
	4.93	4.82	4.61	5.40	5.24
MobileNet + 900 epoch	100	200	300	400	500
	3.89	3.83	3.94	4.01	4.12
	600	700	800	900	1000
	4.23	4.09	4.41	4.56	4.52
ResNet50 + 1 Dense layer + 400 epoch	100	200	300	400	500
	1.44	3.27	3.45	3.45	3.63
	600	700	800	900	1000
	3.79	3.85	4.02	4.23	4.55
NASNetMobile + 2 Dense layer + 900 epoch	100	200	300	400	500
	3.88	2.27	2.39	2.50	2.64
	600	700	800	900	1000
	2.59	2.70	2.66	2.82	2.68

(표 8) atan함수 이용하고 학습에 32 * 32 확장한 이미지를 사용한 모델들의 FID 값

(Table 8) FID per epoch of each optimized 5 models trained by images which are expanded to 32 * 32 using atan function.

model	FID by epoch				
DenseNet121 + 1 Dense layer + 700 epoch	100	200	300	400	500
	0.271	0.079	0.063	0.054	0.059
	600	700	800	900	1000
	0.059	0.059	0.053	0.076	0.055
VGG19 + 2 Dense layer + 100 epoch	100	200	300	400	500
	0.171	0.095	0.117	0.117	0.127
	600	700	800	900	1000
	0.139	0.123	0.114	0.156	0.144
MobileNet + 900 epoch	100	200	300	400	500
	0.368	0.10	0.094	0.102	0.103
	600	700	800	900	1000
	0.109	0.105	0.124	0.136	0.124
ResNet50 + 1 Dense layer + 400 epoch	100	200	300	400	500
	0.163	0.087	0.091	0.095	0.094
	600	700	800	900	1000
	0.105	0.098	0.101	0.110	0.123
NASNetMobile + 2 Dense layer + 900 epoch	100	200	300	400	500
	0.659	0.063	0.051	0.056	0.064
	600	700	800	900	1000
	0.064	0.067	0.058	0.079	0.059

3.2.2 $y = atan(x)$를 이용하여 32 * 32 이미지를 학습 후 평가 결과

3.2.1에서 기준으로 잡은 모델에 대해서 3.1.6에서 증진된 이미지를 사용하여 epoch 별로 IS 와 FID를 측정해보았다. (표 7), (표 8), (표 9), (표 10), (표 11), (표 12)에서는 IS 와 FID를 통해 각 모델마다 가장 성능이 좋은 값에 노란색으로 표시를 하였다.

3.2.2의 IS 측정 결과는 (표 7), FID 측정 결과는 (표 8)에서 확인할 수 있다. 가장 높은 IS 값은 VGG19의 5.40이며, 가장 낮은 FID는 NASNetMobile 의 0.051이다. IS와 FID는 비례 및 반비례 관계를 보이지 않았다. IS 로 가장 생성이 잘 됐다고 판단되는 epoch 구간은 대개 900 ~ 1000 epoch이며, FID는 200 ~ 300 epoch이다.

(표 9) tanh함수 이용하고 학습에 32 * 32 확장한 이미지를 사용한 모델들의 IS 값

(Table 9) IS per epoch of each optimized 5 models trained by images which are expanded to 32 * 32 using tanh function.

model	IS by epoch				
DenseNet121 + 1 Dense layer + 700 epoch	100	200	300	400	500
	1.00	2.22	2.73	2.85	3.15
	600	700	800	900	1000
	3.31	3.28	3.29	3.40	3.39
MobileNet + 900 epoch	100	200	300	400	500
	1.28	1.69	1.97	2.02	2.32
	600	700	800	900	1000
	2.49	2.49	2.54	2.67	2.74
ResNet50 + 1 Dense layer + 400 epoch	100	200	300	400	500
	2.51	3.46	3.75	3.87	4.17
	600	700	800	900	1000
	4.23	4.50	4.41	4.83	5.04
NASNetMobile + 2 Dense layer + 900 epoch	100	200	300	400	500
	2.39	2.81	3.05	3.03	3.21
	600	700	800	900	1000
	3.28	3.46	3.37	3.31	3.36

(표 10) tanh함수 이용하고 학습에 32 * 32 확장한 이미지를 사용한 모델들의 FID 값

(Table 10) FID per epoch of each optimized 5 models trained by images which are expanded to 32 * 32 using tanh function.

model	FID by epoch				
DenseNet121 + 1 Dense layer + 700 epoch	100	200	300	400	500
	0.434	0.078	0.060	0.063	0.071
	600	700	800	900	1000
	0.073	0.064	0.061	0.064	0.066
MobileNet + 900 epoch	100	200	300	400	500
	0.232	0.100	0.071	0.073	0.065
	600	700	800	900	1000
	0.065	0.063	0.063	0.062	0.055
ResNet50 + 1 Dense layer + 400 epoch	100	200	300	400	500
	0.384	0.091	0.082	0.087	0.098
	600	700	800	900	1000
	0.107	0.104	0.089	0.107	0.113
NASNetMobile + 2 Dense layer + 900 epoch	100	200	300	400	500
	0.596	0.096	0.086	0.077	0.091
	600	700	800	900	1000
	0.092	0.085	0.083	0.083	0.090

3.2.3 $y = \tanh(x)$를 이용하여 32 * 32 이미지를 학습 후 평가 결과

3.2.2와 모든 조건은 동일하게 하되, 전단사화 함수를 atan에서 tanh로 변경하여 새로운 이미지를 만든 뒤, 3.2.1에서 기준으로 잡은 모델에 대해서 학습을 진행하여 IS, FID를 비교해보았다.

3.2.3의 IS 측정 결과는 (표 9), FID 측정 결과는 (표 10)에서 확인할 수 있다.가장 높은 IS 값은 ResNet50의 5.04이며, 가장 낮은 FID는 MobileNet의 0.055이다. VGG19 같은 경우, [1.0, 0, 0 ... 0] 형태로 일정한 형태의 확률 벡터만 생성이 되어 IS 와 FID 측정이 불가능하였다. IS 로 가장 생성이 잘 됐다고 판단되는 epoch 구간은 대개 900 ~ 1000 epoch이며, FID는 300 ~ 400 epoch 이다.

(표 11) atan함수 이용하고 학습에 8 * 8 이미지를 사용한 모델들의 IS 값

(Table 11) IS per epoch of each optimized 5 models trained by images which are expanded to 8 * 8 using atan function.

model	IS by epoch				
DenseNet121 + 1 Dense layer + 700 epoch	100	200	300	400	500
	1.05	6.80	7.40	7.16	7.63
	600	700	800	900	1000
	8.27	7.73	7.66	7.89	7.02
MobileNet + 900 epoch	100	200	300	400	500
	2.28	5.48	7.50	6.97	7.44
	600	700	800	900	1000
	7.26	7.28	7.11	7.47	7.51
ResNet50 + 1 Dense layer + 400 epoch	100	200	300	400	500
	1.34	4.74	5.63	5.29	6.14
	600	700	800	900	1000
	6.19	6.48	6.41	6.63	6.68
NASNetMobile + 2 Dense layer + 900 epoch	100	200	300	400	500
	2.01	3.01	6.67	6.79	7.85
	600	700	800	900	1000
	7.81	8.27	8.41	8.64	8.34

3.2.4 $y = atan(x)$를 이용하여 8 * 8 이미지를 학습 후 평가 결과

3.2.2 와 모든 조건은 동일하게 하되, 이미지를 32 * 32로 확장 변환해서 학습시키지 않고 8 * 8 그대로 이용하여 3.2.1에서 기준으로 잡은 모델에 대해서 학습을 진행하여 IS, FID를 비교해보았다. 이 경우, (8,8,3)를 입력을 받아 (32,32,3)를 출력하는 덴스 레이어를 각 모델 앞에 추가하여 8 * 8 크기를 입력으로 받을 수 있도록 하였다.

3.2.4의 IS 측정 결과는 (표 11), FID 측정 결과는 (표12)에서 확인할 수 있다. 가장 높은 IS 값은 NASNetMobile의 8.64이며, 가장 낮은 FID는 DenseNet121의 0.150이다. VGG19 같은 경우, 확률 벡터가 3.2.3 경우와 동일하게 생성이 되어 IS 와 FID 측정이 불가능하였다. IS 로 가장 생성이 잘 됐다고 판단되는 epoch 구간은 대개 900 ~ 1000 epoch 이며, FID 는 1000 epoch 이다.

(표 12) atan함수 이용하고 학습에 8 * 8 이미지를 사용한 모델들의 FID 값

(Table 12) FID per epoch of each optimized 5 models trained by images which are expanded to 8 * 8 using atan function.

model	FID by epoch				
DenseNet121 + 1 Dense layer + 700 epoch	100	200	300	400	500
	0.457	0.340	0.486	0.303	0.312
	600	700	800	900	1000
	0.304	0.220	0.201	0.213	0.150
MobileNet + 900 epoch	100	200	300	400	500
	0.680	0.301	0.308	0.206	0.209
	600	700	800	900	1000
	0.181	0.172	0.156	0.174	0.152
ResNet50 + 1 Dense layer + 400 epoch	100	200	300	400	500
	0.332	0.491	0.265	0.177	0.201
	600	700	800	900	1000
	0.180	0.183	0.170	0.174	0.152
NASNetMobile + 2 Dense layer + 900 epoch	100	200	300	400	500
	1.073	1.058	0.641	0.388	0.380
	600	700	800	900	1000
	0.360	0.311	0.276	0.279	0.225

4. 결 론

atan 보다 tanh 함수를 사용했을 때 정확도가 대개 3% 내외로 증가했다. IS의 경우, MobileNet은 절반에 가깝게 감소하고 그 외에는 10% 내외로 증가했으나, 큰 변화는 없었다. 그 반면에 FID는 MobileNet은 60% 정도 값이 감소하여 더 좋은 결과를 보이고. NASNetMobile 은 50% 정도 값이 증가하였다. 그 외에는 크게 변하지 않았다. 즉, atan 대신에 tanh를 사용하면 IS를 약간 증가 시키는 효과를 볼 수 있으나 FID에서는 좋은 효과를 기대하기 어렵다.

(표 13) 각 모델 별 accuracy 값

(Table 13) accuracy value by model

model	32* 32 atan accuracy (%)	32 * 32 tanh accuracy (%)	8 * 8 atan accuracy (%)
DenseNet121	80.22	82.31	92.93
VGG19	77.86	65.88	72.80
MobileNet	79.18	81.95	91.46
ResNet50	78.11	78.92	93.46
NASNetMobile	79.91	83.56	92.54

32 * 32 이미지보다 8 * 8 이미지를 사용하게 되면 정확도가 10% 내외로 좋아지나 VGG19는 반대로 감소했다. DenseNet121, MobileNet, ResNet50의 IS는 최소 1.5배 이상 좋아지고 NASNetMobile의 IS는 오히려 감소했다. FID에서는 적어도 2배에서 3배 이상 값이 증가하며 평가 성능이 좋지 않아졌다. 32 * 32 이미지 대신 8 * 8 이미지를 사용하게 되면 IS에서는 큰 개선이 보인다. 이 처럼 IS와 FID, accuracy 세 지표가 서로 비례의 관계를 가지지 않기 때문에 특정 모델이 적합하다고 단정 짓기는 어렵다.

그러나 이 와중에도 적합한 모델을 특정 지어보고자 했다. VGG19 모델 (표 4)와 같이 정확도가 낮고 극단적인 확률 벡터로 인해 IS, FID가 구하기 어렵다는 점에 평가 모델로 적합하지 않았다.

3.2.4의 DenseNet121 과 NASNetMobile 이 정확도 및 IS가 모든 모델이 전체적으로 높게 나왔다. 따라서 DenseNet121, NASNetMobile을 atan을 이용하여 8 *8 이미지를 학습 시켜 IS를 평가하는 방법이 제일 적합하다. NASNetMobil 은 어떤 방법에서든 DenseNet121 보다 FID의 결과가 좋지 않으므로 정확도까지 고려해봤을 때 DenseNet121이 가장 평가에 사용되는 적합한 모델로 보인다. FID만을 평가하는 방법은 tanh 보다 atan 이 더 좋으나, 이미지 사이즈가 작아질수록 정확도는 높아지고 FID 수치는 높아지므로 평가자의 평가하고자 하는 가중에 이 따라 이미지 사이즈 결정 하는 것을 권장한다.

참고문헌(Reference)

[1] Nour Moustafa and Jill Slay, “The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data sets” A Global Perspective, Vol. 25, Issue 1-3, pp. 18-31, 2016.

https://doi.org/10.1080/19393555.2015.1125974

[2] Mahbod Taballaee, Ebrahim Bagheri, Wei Lu and Ali A. Ghorbani, “A detailed Analysis of the KDD CUP 99 data set”, 2009 IEEE Symposium on CISDA, pp.53-58. July 2019. https://doi.org/10.1109/CISDA.2009.5356528

[3] Shane Barratt and Rishi Sharma, “A Note on the Inception Score”, arXiv preprint arXiv:1801.01973, 2018. https://arxiv.org/abs/1801.01973

[4] K. Shmelkov, C. Schmid and K. Alahari. “How good is my GAN?” 2018 ECCV pp. 213-229, 2018. http://openaccess.thecvf.com/content_ECCV_2018/html/Konstantin_Shmelkov_How_good_is_ECCV_2018_paper.html

[5] Martin Arjovsky, Soumith Chintala, Leon Bottou, “Wasserstein GAN”, Proceedings of the 34th Intenational Conference on Machine Learning, PMLR, 70:214-223, 2017. http://proceedings.mlr.press/v70/

[6] Nataraj, L., Karthikeyanm, S., Jacob, G., Manjunath, B.S. “Malware images: visualization and automatic classification.” Proceedings of the Conference on Visualizing for Cyber Securty, p. 4, 2011. https://doi.org/10.1145/2016904.2016908

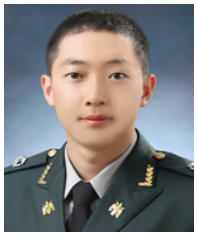
[7] Borjim A. “Pros and cons of GAN evaluation measures.” Computer Vision and Image Understanding, 2019. https://doi.org/10.1016/j.cviu.2018.10.009

[8] S. Revathi, Dr. A. Malathi, “A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection”, International Journal of Engineering Research & Technology (IJERT), Vol. 2 Issue 12, 2013. https://www.ijert.org/volume-02-issue-12-december-2013

[9] S. Choi, S. Jang, Y. Kim, J.Kim, “Malware detection using malware image and deep learning”, International Conference on Information and Communication Technology Convergence (ICTC), pp. 1193-1195, 2017. https://doi.org/10.1109/ICTC.2017.8190895

◐ 저 자 소 개 ◑

박 용 빈(Yong-bin Park)
2018년 고려대학교 사이버국방학과(공학사)
2018년~현재 국방과학연구소 현역연구원 육군 중위
관심분야 : 인공지능, 네트워크, etc.
E-mail : sinen0308@add.re.kr



신 성 욱(Sung-uk Shin)
2017년 고려대학교 사이버국방학과(공학사)
2017년~현재 국방과학연구소 현역연구원 육군 중위
관심분야 : 딥러닝.
E-mail : ssw1419@add.re.kr

이 인 섭(In-sup Lee)
2018년 고려대학교 사이버국방학과(공학사)
2018년~현재 국방과학연구소 현역연구원 육군 중위
관심분야 : ML-based Network Security, Anomaly Detection, Generative Models.
E-mail : dlstjq0711@add.re.kr