https://doi.org/10.33778/kcsa.2023.23.5.057

학습 데이터가 없는 모델 탈취 방법에 대한 분석★

권 현*, 김 용 기**, 이 준***

요 약

딥뉴럴네트워크 모델의 취약점으로 모델 탈취 방법이 있다. 이 방법은 대상 모델에 대하여 여러번의 반복된 쿼리를 통해서 유사 모델을 생성하여 대상 모델의 예측값과 동일하게 내는 유사 모델을 생성하는 것이다. 본 연구에서, 학습 데이터가 없이 대상 모델을 탈취하는 방법에 대해서 분석을 하였다. 생성 모델을 이용하여 입력 데이터를 생성하고 대상 모델과 유사 모델의 예측값이 서로 가까워지도록 손실함수를 정의하여 유사 모델을 생성한다. 이 방법에서 대상 모델의 입력 데이터에 대한 각 클래스의 logit(로직) 값을 이용하여 경사하강법으로 유사 모델이 그것과 유사하도록 학습하는 과정을 갖는다. 실험 환경으로 pytorch 머신러닝 라이브러리를 이용하였으며, 데이터셋으로 CIFAR10과 SVHN을 사용하였다. 대상 모델로 ResNet 모델을 이용하였다. 실험 결과로써, 모델 탈취 방법은 CIFAR10에 대해서 86.18%이고 SVHN에 대해서 96.02% 정확도로 대상 모델과 유사한 예측값을 내는 유사 모델을 생성하는 것을 볼 수가 있었다. 추가적으로 모델 탈취 방법에 대한 고려사항와 한계점에 대한 고찰도 분석하였다.

Analysis of methods for the model extraction without training data

Hyun Kwon*, Yonggi Kim**, Jun Lee***

ABSTRACT

In this study, we analyzed how to steal the target model without training data. Input data is generated using the generative model, and a similar model is created by defining a loss function so that the predicted values of the target model and the similar model are close to each other. At this time, the target model has a process of learning so that the similar model is similar to it by gradient descent using the logit (logic) value of each class for the input data. The tensorflow machine learning library was used as an experimental environment, and CIFAR10 and SVHN were used as datasets. A similar model was created using the ResNet model as a target model. As a result of the experiment, it was found that the model stealing method generated a similar model with an accuracy of 86.18% for CIFAR10 and 96.02% for SVHN, producing similar predicted values to the target model. In addition, considerations on the model stealing method, military use, and limitations were also analyzed.

Key words : Model extraction, Deep neural network, Adversarial example, ResNet

접수일(2023년 05월 08일), 수정일(2023년 05월 24일), 게재확정일(2023년 10월 10일)

★ 본 논문은 육군사관학교 미래전략기술연구소의 2023년(23-AI-연구소-04) 저술활동비 지원을 받아 연구되었음..

* 육군사관학교 AI·데이터과학과 부교수(주저자)

** 과학기술정책연구원 과학기술외교안보연구단 부연구위원

*** 호서대학교 게임소프트웨어학과 교수(교신저자)

1. 서 론

지도학습 기반에 분류 문제에 있어서 딥뉴럴네트워크는 좋은 성능을 제공한다. 딥뉴럴네트워크는 이미지, 음성, 텍스트 등에 좋은 성능을 제공하지만 그 중에서 이미지 분야에 있어서 특히 좋은 성능을 제공한다[1]. 2017년 ImageNet 챌린지[2]에서는 사람이 이미지를 분류하는 것 보다 합성곱 신경망이 이미지를 분류하는 것이 더 성능이 좋게 나와서 더 이상 챌린지 대회를 하지 않고 있다. 이러한 합성공 신경망[3]을 기반으로 한 딥뉴럴네트워크를 이용하여 마스크 탐지, 얼굴 인식, 신분증 인식 등 이미지 인식 및 분류에서 실생활에서 상용화되고 있다.

하지만 이러한 딥뉴럴네트워크는 취약점이 존재한다. 딥뉴럴네트워크의 취약점으로 모델 인식 측면에서 적대적 샘플[4-6], 중독 샘플 공격[7], 백도어 공격[8] 등으로 데이터를 조작하거나 악성 데이터를 학습하여 모델의 성능을 저하시키는 방법이 있다. 모델나 데이터 측면에서, 대상 모델에 대한 정보를 추출하여 유사한 모델을 생성하여 모델을 탈취[9]하거나 모델이 학습한 데이터를 탈취하는 방법이 있다. 이러한 딥뉴럴네트워크의 취약점에 대한 다양한 연구가 진행되고 있으며 본 연구에서는 모델 탈취 연구에 대해서 분석하고자 한다.

모델 탈취 연구[9-10]를 하기 위해서, 대상 모델에 대한 학습데이터를 알고 있고 대상 모델에 대해서 입력을 주었을 때, 입력 값에 대한 각 확률값에 대한 정보가 필요하다. 왜냐하면 학습 데이터를 유사 모델과 대상 모델에게 둘 다 입력 한 후에 유사 모델에서 나온 확률값과 대상 모델에서 나온 확률값이 같도록 손실함수를 구성하여 그것을 최소화하는 방식으로 유사 모델을 학습하는 과정을 갖는다. 그러면 유사 모델이 대상 모델과 유사한 파라미터로 최적화 되어 대상 모델을 탈취가 가능하다. 하지만 최근에 학습 데이터에 대한 정보가 없이 대상 모델에 대한 확률값만 알고도 모델 탈취하는 연구가 나오게 되었으며 그것에 대한 각종 파라미터에 따른 성능 분석을 본 연구에서 하고자 한다.

본 연구에서 우리는 학습데이터에 대한 정보가 없이 모델을 탈취하는 연구에 대한 다양한 분석을 하였다. 이 연구에서 학습 데이터의 정보가 없이 노이즈로부터 데이터를 생성한 후에 대상 모델과 유사 모델에 제공하여 각각의 손실함수를 구성하여 유사 모델을 대상 모델과 같게 학습하는 방법이다. 본 연구에서 공헌점은 다음과 같다. 먼저, 학습데이터 없이 모델을 탈취하는 방법에 대하여 각 파라미터가 미치는 영향에 대해서 분석을 하였다. 파라미터로 학습률, 손실값, logit 교정의 평균, 노름 방식 등에 대한 시스템의 성능을 분석하였다. 두 번째로 데이터셋으로 CIFAR10[11]과 SVHN[12] 데이터셋을 통하여 성능을 검증하였다. 세 번째로, 대상 모델로 ResNet 모델[13]에 대하여 비교 성능 분석을 하였다.

이 장의 나머지 구성은 다음과 같다. 2장에서 관련연구에 대한 소개를 하고 3장에서 모델 탈취 방법에 대한 설명을 한다. 4장에서 실험 환경 및 실험결과를 소개하고 연구에 대한 고찰을 다룬다. 마지막 5장에서 이 논문의 결론으로 구성되어 있다.

2. 관련연구

학습 데이터가 없이 모델 탈취 방법은 데이터가 없는 지식 증류 방법[14]에서 비롯되었다. 데이터가 없는 지식 증류방법은 선생 모델(teacher model)이 학생 모델(student model)에게 지식을 전송하는 방법이다. 이 방법은 통상적으로 계산양이 한계가 있는 환경에서 모델의 사이즈를 줄이기 위해서 사용되는 방법이다. 지식 증류를 하기 위해서 선생 모델을 학습했던 데이터셋에 대한 정보는 너무 크거나 비밀성이 있어서 공개되지 않은 경우가 있기 때문에 데이터가 없는 지식 증류방식이 사용된다. 데이터가 없는 지식 증류방법은 학생 모델을 선생 모델처럼 만들기 위한 선생 모델에 제공할 데이터를 합성하는 생성 모델을 훈련시킨다.

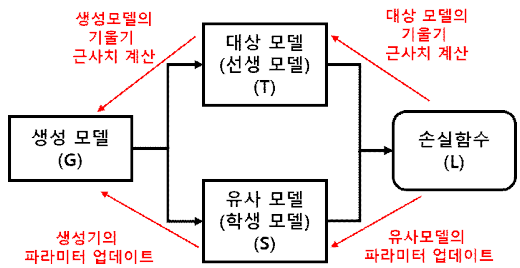
위와 같이 유사하게 학습 데이터가 없는 모델 추

출방식을 하기 위해서 학생 모델이 선생 모델에게 쿼리하는 학습 데이터를 생성하는 것이 필요하다. 본 연구에서 적대적 생성 모델 (adversarial generative net)[15]에서 생성 모델과 분류 모델을 제로섬 게임처럼 서로 학습하는 과정을 응용한다. 생성 모델에서 임의로 생성한 데이터를 선생 모델과 학생 모델에게 제공하고 그 결과값에 대한 차이를 가장 잘 나타내는 데이터를 생성하도록 학습하는 과정을 갖는다. 반면에 학생 모델은 선생 모델과 유사한 예측값이 나오도록 학생 모델의 파라미터를 학습하는 과정을 가진다.

따라서 본 연구에서 학습 데이터가 없는 모델 탈취 방법은 데이터가 없는 지식 증류 방법과 적대적 생성 모델을 응용하여 합성 데이터를 생성하면서 동시에 학생 모델(유사 모델)을 선생 모델(대상 모델)처럼 유사하게 예측값을 만들도록 학생 모델을 학습하는 과정을 가진다. 이에 대한 자세한 내용은 3장에서 설명하였다.

3. 모델 탈취 방법

(그림 1)은 모델 탈취 방법의 구조도를 보여준다. 대상 모델(선생 모델)과 유사한 예측값을 제공하는 유사 모델(학생 모델)을 생성하는 것이 제안 방법의 목표이다. 이를 하기 위해서, 생성 모델은 평균 0이고 표준편차가 1인 노이즈를 입력으로 받아 입력데이터를 생성한다. 이렇게 생성된 입력데이터는 대상 모델과 유사 모델에게 입력으로 제공하여 이에 대한 각 클래스의 클래스점수(logit value)값을 출력하고 이 logit value를 이용하여 손실함수를 구성한다. 이 logit value이 L1 노름을 이용하여 서로 최대한 값이 같도록 손실함수를 구성한 후에 손실함수를 최소화할수록 대상 모델의 예측값과 유사 모델의 예측값이 같아지도록 유사 모델의 파라미터를 업데이트를 한다.



(그림 1) 모델 탈취 방법의 구조

반면에 생성모델에서 출력되는 입력데이터는 대상 모델과 유사 모델의 예측값이 최대한 다르게 나오도록 입력값이 생성되도록 학습하는 과정을 갖습니다. 이는 마치 적대적 생성 모델 방식으로 생성 모델에서 주는 입력 데이터는 대상 모델과 유사 모델의 예측값이 크게 나오도록 입력값을 생성하고 반면에 유사모델은 대상 모델의 예측값과 유사하도록 모델의 파라미터를 학습합니다. 이러한 과정을 계속 반복하게 되면 유사 모델은 대상 모델과 유사하게 예측하는 모델을 생성할 수가 있다.

이에 대한 수학적인 표현으로 나타내면 다음과 같다.

$$\min_S \max_G E_{z \sim N(0,1)} \left[L\left(V\left(G(z;\theta_G) \right), S\left(G(z;\theta_G) \right) \right) \right]$$

여기서, 노이즈 $z \sim N(0,1)$ 에 대해서 평균이 0이고 분산이 1인 정규분포에서 추출한 후에 생성 모델에 입력으로 제공하여 입력 데이터 $x = G(z;\theta_G)$ 를 생성한다. 여기서 $G(\bullet)$ 는 생성 모델의 동작 함수를 의미하며 θ_G 를 G 모델의 파라미터를 의미한다. $V(\bullet)$ 와 $S(\bullet)$ 는 대상 모델(선생 모델)의 동작함수와 유사 모델(학생 모델)의 동작 함수이다.

$L\left(V\left(G(z;\theta_G) \right), S\left(G(z;\theta_G) \right) \right) = L\left(V(x), S(x) \right)$ 는 손실함수로서 입력 데이터 $x = G(z;\theta_G)$ 에 대하여 $V(x)$ 모델의 예측값과 $S(x)$ 모델의 예측값에 대한 L1 노름을 의미한다. 수식적으로 표현하면 x 의 성분이 K개라고 했을 때 $L\left(V(x), S(x) \right) = \sum_{i=1}^{K} \left| v_i - s_i \right|$ 를 의미한다. 생성 모델 G는 손실함수의 평균값을 극대화하는 방향으로 모델이 학습하고 반대로 유사 모델 S는 손

실함수의 평균값이 최소화 하도록 각 모델의 파라미터를 경사하강법을 이용하여 학습 진행한다. 위 과정을 반복하게 되면 유사 모델의 예측값은 대상 모델의 예측값과 유사하게 되어 대상 모델을 탈취할 수 있게 된다.

4. 실험 및 평가

제안 방법을 구현하기 위해서 실험환경으로 pytorch 머신러닝 라이브러리[16]를 사용하였으며, 서버는 Intel(R) Core(TM) i3-7100 CPU @ 3.90GHz와 GPU는 GeForce GTX 1050을 사용하였다.

4.1 데이터셋

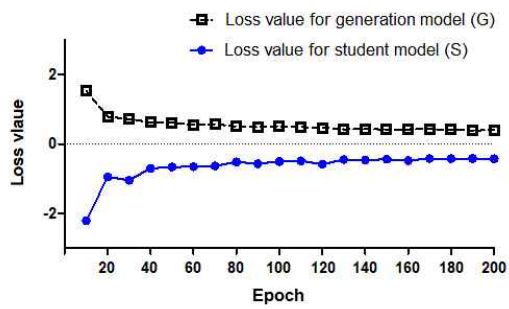
데이터셋은 CIFAR10과 SVHN 데이터셋으로 구성되어 있다. CIFAR10은 컬러 이미지 데이터셋으로 자동차, 고양기, 개, 사슴, 개구리, 말 등에 사물 객체로 10가지 종류로 구성되어 있다. 이미지 사이즈는 가로 32, 세로 32, 채널 3으로 되어 있으며 훈련데이터셋은 6만개로 구성되어 있으며 테스트 데이텃세은 1만개로 구성되어 있다. SVHN 데이터셋은 컬러이미지 숫자 데이터셋으로 0부터 9로 10가지 종류로 구성되어 있다. 이미지 사이즈는 가로 32, 세로 32, 채널 3으로 되어 있으며 훈련데이터셋은 73,257개가 있고 테스트 데이터셋은 26,032개로 구성되어 있다.

4.2 대상 모델과 유사 모델

대상 모델 및 유사 모델 구성은 합성곱 신경망을 기본으로 하였다. 대상 모델은 Resnet 모델로 34개 층으로 구성된 모델을 하였다. 반면에 유사 모델은 18층으로 구성된 Resnet 모델로 구성하였다. 기본 배치사이즈는 256으로 하였고 쿼리는 200만번을 하였고 epoch은 50을 하였다. 생성 모델의 학습률은 0.0005로 하였고 유사 모델의 학습률은 0.1로 하였다. 학습 스케쥴러는 multistep으로 하였고 0.5와 0.8로 잡았고 크기는 0.3으로 하였다. 기본적인 하이퍼 파라미터이고 이에 대한 하이퍼 파라미터 변경에 따른 성능 분석을 4.3 실험결과에 분석하였다.

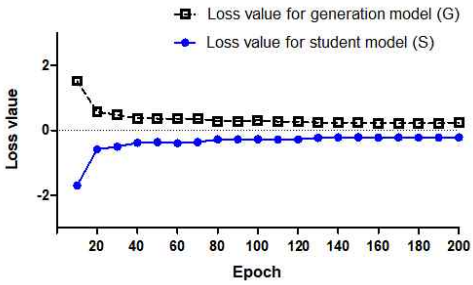
4.3 실험결과

이 소챕터에서 CIFAR10과 SVHN 데이터셋에 대한 모델 탈취에 대한 성능을 epoch에 따라 정확도(accuracy), 노름 기울기값(norm gradient), 손실값(loss value)의 분석한 결과를 보여준다.
(그림 2)는 CIFAR10에 대하여 epoch에 따른 손실값(loss value)를 보여준다. 한 epoch 당 50번에 iteration이 반복된다. epoch이 증가할수록 생성모델의 손실값과 유사 모델(학생 모델, student model)의 손실값이 작아지는 것을 볼 수가 있다. 이를 통해서 생성 모델(generation model)에서 제공하는 출력값은 대상 모델과 유사 모델의 예측값이 크게 벌어지도록 하는 특징이 잘 반영되도록 학습된 것을 볼 수가 있다. 반면에, 유사 모델(학생 모델)은 대상 모델의 예측값과 유사하게 예측값을 출력하도록 잘 학습 된 것을 볼 수가 있다.



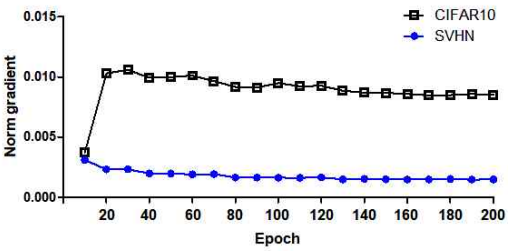
(그림 2) CIFAR10에 대한 epoch에 따른 손실함수

(그림 3)는 SVHN에 대하여 epoch에 따른 손실값(loss value)를 보여준다. 그림에서, epoch에 따라 생성 모델과 유사 모델(학생 모델)의 손실값이 작아지는 것을 볼 수가 있다. 특히, (그림 2)의 CIFAR10에 비해 SVHN에서 손실값이 더 작게 잘 학습된 것을 볼 수가 있다.

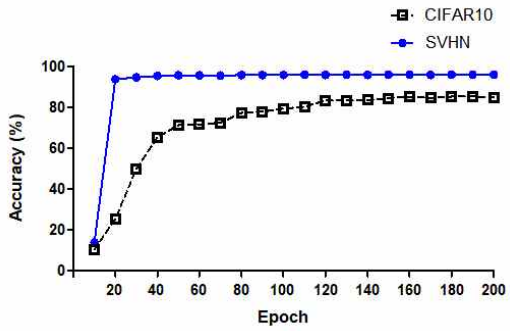


(그림 3) SVHN에 대한 epoch에 따른 손실값

(그림 4)는 epoch에 따른 L1 노름 기울기값의 변화를 CIFAR10과 SVHN에 대한 수치를 보여준다. 그림에서, epoch에 따라서 L1 노름 기울기값이 작아지는 것을 볼 수가 있다. 이는 대상 모델의 예측값과 유사 모델의 예측값의 차이가 작은 것으로 유사 모델이 대상 모델과 거의 유사한 예측값을 출력한 것을 볼 수가 있다. 특히, CIFAR10 데이터셋보다 SVHN 데이터셋에서 좀 더 좋은 성능으로 유사 모델의 예측값이 대상 모델의 예측값의 차이가 거의 없는 것을 볼 수가 있다.

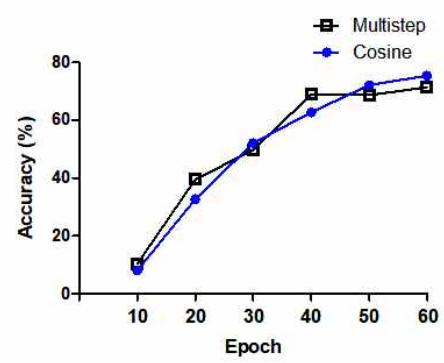


(그림 4) CIFAR10과 SVHN 데이터셋에서 epoch에 따른 L1 노름 경사값에 변화



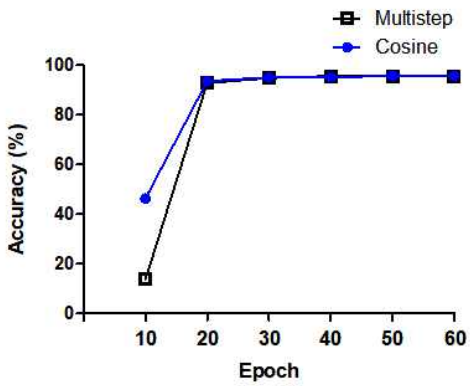
(그림 5) epoch에 따른 유사 모델의 정확도(accuracy)

(그림 5)는 epoch에 따른 유사 모델의 정확도를 보여준다. epoch이 증가할수록 유사 모델의 정확도가 향상되는 것을 볼 수가 있다. epoch이 200일 때, SVHN 데이터셋에 대하여 96.02% 정확도를 가지고 CIFAR10 데이터셋에 대하여 86.18% 정확도를 가진다.



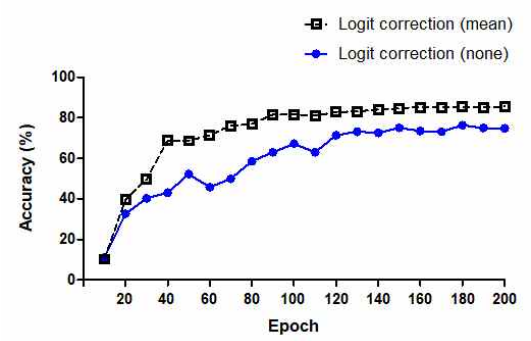
(그림 6) CIFAR10에서 학습률 방식인 multistep과 consine 방식에 에 대한 epoch에 따른 유사모델의 정확도(accuracy)

(그림 6)는 CIFAR10에서 epoch에 따른 유사 모델의 정확도를 보여준다. epoch이 증가할수록 정확도가 향상된 것을 볼 수가 있다. multistep과 cosine 방식 둘다 유사한 성능으로 잘 학습되는 것을 볼 수가 있다.



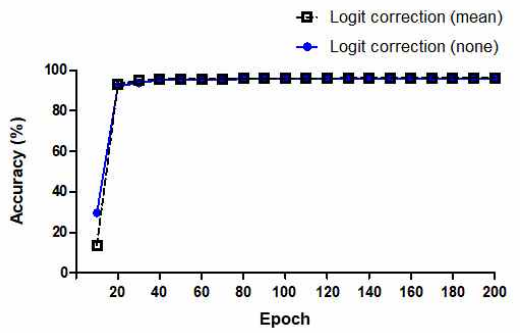
(그림 7) SVHN에서 학습률 방식인 multistep과 consine 방식에 대한 epoch에 따른 유사모델의 정확도(accuracy)

(그림 7)는 SVHN에서 epoch에 따른 유사 모델의 정확도를 보여준다. epoch이 증가할수록 정확도가 향상된 것을 볼 수가 있다. (그림 6)과 비슷하게 multistep과 cosine 방식 둘다 유사한 성능으로 잘 학습되고 epoch 20 이후부터는 거의 정확도가 약 96%으로 일정하게 유지되는 것을 볼 수가 있다.



(그림 8) CIFAR10에서 logit correction 방식에 따른 epoch에 따른 정확도(accuracy)

(그림 8)는 CIFAR10에서 epoch에 따른 logit correction 방식에 따른 정확도를 보여준다. epoch이 증가할수록 정확도가 향상된 것을 볼 수가 있다. logit correction에서 평균값으로 교정한 것이 하지 않은 것에 비해 정확도 성능이 좋은 것을 볼 수가 있었다.



(그림 9) SVHN에서 logit correction 방식에 따른 epoch에 따른 정확도(accuracy)

(그림 9)는 SVHN에서 epoch에 따른 logit correction 방식에 따른 정확도를 보여준다. epoch이 증가할수록 정확도가 향상된 것을 볼 수가 있다. 1 ogit correction에서 평균값과 하지 않은 것에 대한 결과값이 거의 유사한 것을 볼 수가 있다. 오히려 초반 epoch의 경우, 평균 logit correction 보다 하지 않은 것이 더 좋은 정확도가 나온 것을 볼 수가 있었다.

4.4 분석 및 논의

제안 방법에서 적대적 생성 모델 및 선생 모델과 학생 모델을 통한 지식 전달 방식을 두가지 동시에 사용하는 특징이 있다. 기본적으로 적대적 생성 모델의 제로섬 개념을 이용하지만 두 개 모델에 대한 피드백을 통해서 입력 데이터를 생성한다.

제안 방법의 제한사항으로, 블랙박스에서 유사 모델을 생성하기 위해서 200만번의 반복된 쿼리가 필요하다. 대상 모델의 경우, 반복된 쿼리가 제한되는 모델일 경우 유사 모델 생성이 제한이 될 수가 있다. 또한, 경사하강법에 의해서 모델의 파라미터를 학습하는 과정이기 때문에 기울기 손실현상이 일어날 수가 있으며 초기값 설정에 따른 성능 변화 등에 불완전한 부분이 있을 수 있다.

5. 결 론

본 연구에서 우리는 학습데이터에 대한 정보가

없이 모델을 탈취하는 연구에 대한 다양한 분석을 하였다. 이 연구에서 학습 데이터의 정보가 없이 노이즈로부터 데이터를 생성한 후에 대상 모델과 유사 모델에 제공하여 각각의 손실함수를 구성하여 유사 모델을 대상 모델과 같게 학습하는 방법이다. 실험결과로 제안 방법은 CIFAR10과 SVHN 데이터셋에 대하여 각각 86.18%과 96.02% 성능으로 유사 모델을 생성 가능한 것을 볼 수가 있었다.

향후 연구에서는 이미지 데이터셋 뿐만 아니라 음성 데이터셋 또는 텍스트 데이터셋으로 확장해서 연구를 진행할 수가 있다. 또한 제안 방법에 대한 방어연구도 흥미로운 연구주제가 될 것이다.

참고문헌

[1] Li, Zewen, et al. "A survey of convolutional neural networks: analysis, applications, and prospects." IEEE transactions on neural networks and learning systems (2021).

[2] Shankar, Vaishaal, et al. "Evaluating machine accuracy on imagenet." International Conference on Machine Learning. PMLR, 2020.

[3] 이철희, et al. "말벌 영상인식을 위한 심층 합성곱 신경망의 성능 평가." Journal of Apiculture 34.3 (2019): 207-215.

[4] Ko, Kyoungmin, SungHwan Kim, and Hyun Kwon. "Multi-targeted audio adversarial example for use against speech recognition systems." Computers & Security 128 (2023): 103168.

[5] Kwon, Hyun. "Adversarial image perturbations with distortions weighted by color on deep neural networks." Multimedia Tools and Applications 82.9 (2023): 13779-13795.

[6] Kwon, Hyun, and Seung-Hun Nam. "Audio adversarial detection through classification score on speech recognition systems." Computers & Security 126 (2023): 103061.

[7] KWON, Hyun, and Sunghwan CHO. "Multi-Targeted Poisoning Attack in Deep Neural Networks." IEICE TRANSACTIONS on Information and Systems 105.11 (2022): 1916-1920.

[8] Kwon, Hyun. "Multi-model selective backdoor attack with different trigger positions." IEICE TRANSACTIONS on Information and Systems 105.1 (2022): 170-174.

[9] 전소희, et al. "AI 모델 탈취 공격 및 방어 기법들에 관한 연구." 한국정보처리학회 학술대회논문집 28.1 (2021): 382-384.

[10] Truong, Jean-Baptiste, et al. "Data-free model extraction." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021.

[11] Abouelnaga, Yehya, et al. "Cifar-10: Knn-based ensemble of classifiers." 2016 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, 2016.

[12] Maini, Pratyush, Mohammad Yaghini, and Nicolas Papernot. "Dataset inference: Ownership resolution in machine learning." arXiv preprint arXiv:2104.10706 (2021).

[13] Targ, Sasha, Diogo Almeida, and Kevin Lyman. "Resnet in resnet: Generalizing residual architectures." arXiv preprint arXiv:1603.08029 (2016).

[14] Abbasi, Sajjad, et al. "Modeling teacher-student techniques in deep neural networks for knowledge distillation." 2020 International Conference on Machine Vision and Image Processing (MVIP). IEEE, 2020.

[15] Creswell, Antonia, et al. "Generative adversarial networks: An overview." IEEE signal processing magazine 35.1 (2018): 53-65.

[16] Imambi, Sagar, Kolla Bhanu Prakash, and G. R. Kanagachidambaresan. PyTorch." Programming with TensorFlow: Solution for Edge Computing Applications (2021): 87-104.

〔저자소개〕

권 현 (Hyun Kwon)
2010년 2월 육군사관학교 이학사
2015년 8월 KAIST 전산학부 공학석사
2020년 2월 KAIST 전산학부 공학박사
email : hkwon.cs@gmail.com

김용기 (Yonggi Kim)
2007년 8월 포항공대 산업경영공학과 학사
2010년 5월 KDI 국제 정책대학원 경영학 석사, 공공정책학 석사
2022년 2월 서울대학교 환경 계획학 박사
email : rocker9018@gmail.com

이 준 (Jum Lee)
2004년 2월 건국대학교 컴퓨터공학부 공학사
2006년 2월 건국대학교 컴퓨터정보통신공학과 공학석사
2012년 2월 건국대학교 신기술융합학과 공학박사
email : jun.lee.mistra@gmail.com