Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.34, NO.3, Jun. 2024 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2024.34.3.379

딥러닝을 이용한 부채널 데이터 압축 프레임 워크*

정 상 윤,[1†] 진 성 현,[3] 김 희 석[2‡]
[1,2]고려대학교 (대학원생, 교수), [3]삼성전자(연구원)

Side-Channel Archive Framework Using Deep Learning-Based Leakage Compression*

Sangyun Jung,[1†] Sunghyun Jin,[3] Heeseok Kim[2‡]
[1,2]Korea University (Graduate student, Professor), [3]Samsung Electronics (Researcher)

요 약

데이터의 급속한 증가와 함께 저장 공간 절약과 데이터 전송의 효율성이 중요한 문제로 대두되면서, 데이터 압축 기술의 효율성 연구가 중요해졌다. 무손실 알고리즘은 원본 데이터를 정확히 복원할 수 있지만, 압축 비율이 제한적이며, 손실 알고리즘은 높은 압축률을 제공하지만 데이터의 일부 손실을 수반한다. 이에 딥러닝 기반 압축 알고리즘, 특히 오토인코더 모델이 데이터 압축 분야에서 활발한 연구가 진행됐다. 본 연구에서는 오토인코더를 활용한 새로운 부채널 분석 데이터 압축기를 제안한다. 제안하는 부채널 데이터 대상 압축기는 부채널데이터 특성을 잘 유지할 뿐만 아니라, 기존의 널리 사용되는 Delfate 압축방식 대비 높은 압축률을 보인다. 로컬 연결 레이어를 사용한 인코더는 부채널 데이터의 시점별 특성을 효과적으로 보존하고, 디코더는 멀티 레이어 퍼셉트론을 사용하여 빠른 압축 해제 시간을 유지한다. 상관 전력 분석을 통해 제안된 압축기가 부채널 데이터의 특성을 손실 없이 데이터 압축이 가능을 증명하였다.

ABSTRACT

With the rapid increase in data, saving storage space and improving the efficiency of data transmission have become critical issues, making the research on the efficiency of data compression technologies increasingly important. Lossless algorithms can precisely restore original data but have limited compression ratios, whereas lossy algorithms provide higher compression rates at the expense of some data loss. There has been active research in data compression using deep learning-based algorithms, especially the autoencoder model. This study proposes a new side-channel analysis data compressor utilizing autoencoders. This compressor achieves higher compression rates than Deflate while maintaining the characteristics of side-channel data. The encoder, using locally connected layers, effectively preserves the temporal characteristics of side-channel data, and the decoder maintains fast decompression times with a multi-layer perceptron. Through correlation power analysis, the proposed compressor has been proven to compress data without losing the characteristics of side-channel data.

Keywords: Side-Channel Analysis, Compression, Autoencoder, Deep learning

Received(03. 19. 2024), Modified(04. 26. 2024), Accepted(05. 03. 2024)

* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학 ICT연구센터사업의 연구결과로 수행되었음 (IITP-2024-RS-2022-00164800)

† 주저자, jung2217@korea.ac.kr
‡ 교신저자, 80khs@korea.ac.kr(Corresponding author)

I. 서 론

컴퓨터 및 데이터 저장 기술의 발전으로, 대량의 데이터를 처리하고 저장하는 것이 가능해졌다. 처리 및 저장해야하는 데이터양의 증가에 따라 저장 공간 절약과 데이터 전송의 효율성을 고려하는 것은 매우 중요한 문제가 됐다. 이에 따라 자연스럽게 데이터 압축 기술의 효율성을 연구하는 것이 중요해졌다. 데이터 압축 기술은 크게 손실 및 무손실 알고리즘으로 나뉜다. 무손실 알고리즘은 Deflate[1], gzip[2], FPC[3] 등이 압축 해제 시 원본 데이터와 동일한 데이터를 얻을 수 있는 방법이다. 이러한 방법은 압축 비율에 대한 일정한 제한이 있어 손실 압축 알고리즘 보다 많은 저장 공간과 시간이 소요되는 것이 특징이다. 손실 알고리즘은 SZ[4], ZFP[5], ISABELA[6]와 같이 불필요한 데이터를 제거하고 압축하는 방법으로, 데이터 일부의 손실이 허용되는 이미지[7, 42], 오디오, 비디오[8]와 같은 데이터를 압축하는 데 사용된다. 손실 알고리즘은 무손실 알고리즘 보다 높은 압축률을 가지지만, 압축 해제 후 원본 데이터를 복구할 수 없다는 것이 특징이다. 따라서 의료 영상 및 과학 실험 결과와 같이 데이터의 정확성이 중요시되는 분야에는 손실 알고리즘의 사용이 불가능하다.

이에 많은 연구자들이 기존의 압축 알고리즘의 한계점을 극복하기 위해서 딥러닝 기반의 압축 알고리즘을 연구했다. 컴퓨터 비전, 음성 인식[9], 자연어 처리[10]와 같은 다양한 분야에서 딥러닝을 사용한 연구가 지속적으로 이루어졌다. 그 중에서 데이터 압축을 위한 딥러닝 모델로 오토인코더[11]가 효과적임이 증명됐다. 오토인코더는 인코더와 디코더로 구성된 반지도 학습모델로, 인코더는 입력 데이터를 저차원 표현으로 변환하고, 디코더는 다시 저차원 데이터를 원래 데이터로 재구성하는 역할을 한다. 오토인코더는 이미지의 노이즈 제거[12], 이미지 생성[13], 전이학습[14]과 같은 다양한 분야에서 사용되었다. 특히 이미지 크기 압축[15]에서 뛰어난 효과를 보였다. 오토인코더는 원본 데이터와 재구성된 데이터 사이의 차이를 최소화함으로써 원본 데이터에서 핵심적인 특징들을 추출한다. 이러한 특성을 활용하여 Sento[16]는 최소한의 오류 값으로 60배 더 높은 이미지 압축을 하였고, Momenifiar[17]는 벡터 양자화 오토 인코더를 사용하여 고용량의 난류 데이터를 최대 85배 압축하였다.

압축 기술은 부채널 분석에서도 데이터 전송 효율 증가 및 저장 공간의 효율성을 고려 했을 때 매우 중요한 연구이다. 부채널 분석이란 전자기 방출, 전력, 소음, 타이밍, 및 오류와 같은 개발자가 의도하지 않은 정보를 수집하여 내부 비밀 정보를 얻는 공격이다. 부채널 분석을 통하여 암호 시스템[18]의 비밀키를 얻거나 장치에서 멀웨어를 탐지하는 데 역시 부채널 데이터를 사용할 수 있다[19,20]. 정확한 부채널 분석 결과를 얻기 위해서는 높은 샘플링 레이트로 수집된 대량의 파형이 필요하다. 이러한 파형은 연구자에게 상당 시간의 분석 시간과 데이터 저장 공간을 요구한다. 암호는 입력값의 약간의 변화가 중간 및 최종값에 상당한 변화를 일으키는 혼돈과 확산의 원칙에 기반을 둬 설계되어있다. Deflate와 같은 압축 기술은 데이터의 반복성을 이용하여 데이터를 압축하기 때문에 암호 동작 시 발생하는 전력 소비를 압축할 때는 상대적으로 낮은 압축 비율을 가진다. 그러나 많은 연구자들은 Deflate의 범용성, 빠른 처리 속도, 다른 무손실 알고리즘에 비해 높은 압축률 때문에 아직 Deflate를 사용한다.

본 논문은 기존의 압축 기법의 한계를 극복하기 위해 오토 인코더를 사용하는 새로운 부채널 분석 데이터 압축기를 제안한다. 제안하는 압축기는 부채널 특징을 유지하면서, Deflate에 비해 높은 압축률을 가진다. 특히 부채널 특성의 보존을 위하여 인코더에 로컬 연결 레이어[21]를 사용하여, 각 시점에 대한 부채널적인 특성을 잘 보존할 수 있었다. 디코더는 멀티 레이어 퍼셉트론[22]만을 사용하여 Deflate와 유사하거나 거의 동일한 압축 해제 시간을 유지한다. 본 논문은 이를 증명하기 위해 ChipWhisperer 및 ASCAD에서 수집한 AES를 이용해 Deflate와 압축 및 압축/압축해제 시간을 실험했다. 또한 부채널적인 특성을 유지한 채 압축이 가능하다는 것을 증명하기위해, 압축기를 거친 파형을 대상으로 상관 전력 분석을 시행해, 부채널적인 특성에 대한 손실 없이 압축이 가능하다는 것을 증명하였다.

1.1 관련 연구

딥러닝 기술의 발전으로 여러 연구에서 딥러닝을 활용한 연구 결과가 나오면서 자연스럽게 부채널 분석 분야에도 적용하려는 시도가 있었다. [23]는 마스킹 기술을 적용해도 딥러닝 기반 부채널 분석이 가능하다는 것을 증명했고, [24,25]는 합성곱 신경망

을 사용하면 무작위 지터나 지연이 추가된 장치 또한 분석됨을 확인하였다. [26]에서는 서로 다른 장치에서 딥러닝 기반 부채널 분석이 실행가능함을 증명하였다. 또한 비프로파일링 부채널 분석은 공격 대상 시스템의 사전 정보나 프로파일링 없이 공격이 가능함을 시사하였다
[27,28]에서 제안된 딥러닝 기반 부채널 분석 기법 중 하나인 DDLA는 올바른 키에 따라 라벨이 계산될 때의 학습 정확도와 옳지 않은 키에 따라 계산되는 정확도의 차이를 이용하여 키 추측을 하였다.

비밀 키 추측 이외에도 부채널 파형의 전처리 성능 향상에 대한 연구도 활발하게 이루어지고 있다. [29]는 딥러닝 모델에 의도적으로 노이즈를 삽입하여 주요 성분 분석[30] 및 선형 판별 분석[31]과 유사한 효과를 얻을 수 있었다. 또한 [24,32]는 분석에 필요한 파형이 부족할 때 컨볼루션 네트워크와 적대적 신경망을 이용하여 부채널 분석에 필요한 파형을 증강[33]하는 방법을 제안했다. 그 외에도 신호 정렬[34] 및 파형의 공격 지점을 선택하는 [35,36,37] 등 다양한 전처리 단계에 활용 됐다.

딥러닝 네트워크는 이미지 및 비디오와 같은 데이터를 압축하는 일도 효과적임을 입증했다. 특히 오토인코더는 기존에 사용하던 무손실 압축 알고리즘 보다 높은 압축률을 제공하기 때문에 주목을 받았다.[38] [39,40,41]은 인코딩 과정을 통해 데이터의 특징을 보존하고, 디코딩 과정을 통해 원본과 유사한 압축데이터를 생성하는 오토인코더의 특성을 활용하여 데이터 압축을 연구하였다[42]. 그러나 부채널 분석 분야는 아직 파형의 크기를 압축하는 연구는 존재하지 않으며, 데이터의 효율적인 전달과 저장의 효율성을 고려했을 때 반드시 필요한 연구이다.

II. Background

2.1 부채널 분석

암호 알고리즘이 탑재된 보안 디바이스들은 암호학적으로 안전성이 증명되어 있으나, 암호 알고리즘을 실제 소프트웨어나 하드웨어로 구현하는 과정에서 설계자가 의도하지 않은 정보의 누출, 예를 들어 알고리즘이 동작하는 수행 시간, 소비되는 전력, 발생되는 전자기파 등의 부채널 정보가 발생할 수 있다. 이러한 부채널 정보를 이용하여 비밀정보를 분석하는 기법을 부채널 분석[45]이라 하며, 특히 전력 분석은 기기가 암호화 연산을 하는 과정에서 칩에 발생하는 전력 소모량을 바탕으로 비밀 키를 추출하는 방법이다. 칩에서 암호화 연산 시, 데이터의 헤밍웨이트 값에 의존하여 전력 소비량 및 전자파 방출량이 결정되는데, 이를 통하여 암호 비밀키를 추측할 수 있다. 차분 전력 분석 공격은 헤밍 웨이트의 값에 따른 전력 소비량의 차이를 이용하는 대표적인 공격이다. 암호 연산시 발생하는 전력 혹은 전자파의 값은 특정 비트의 값이 0일 때보다 1일 때 더 많이 발생하는 것을 가정하게 되는데, 이때 발생하는 전력 차이를 통해 분석을 시도한다. 특히 상관 전력 분석[46]은 공격자가 사용한 비밀 키가 연관되어 있는 중간값을 계산하고, 추측한 중간값과 소비전력과의 상관계수를 구해, 그중 가장 큰 값을 갖는 키를 실제 키로 추측하는 방법이다.

2.2 DEFLATE

Deflate 알고리즘[1]은 1996년 RFC 1951로 등록된 압축 방법으로, 주로 Windows에서는 zip 파일로, 리눅스에서는 gzip으로 사용되고 있다. 이 방법은 LZ77 압축 알고리즘[43]과 허프만 코딩[44]을 결합하여 데이터를 압축한다. LZ77 알고리즘은 데이터의 반복되는 특성을 이용하여 반복되는 데이터 패턴의 길이와 패턴이 반복되는 위치를 저장하고 압축하는 방법이다. 고정 크기 버퍼인 슬라이딩 윈도우와 동적으로 변화하는 사전을 사용하여 데이터를 압축한다. 입력 값이 들어오면, 슬라이딩 윈도우 내에서 처리된 데이터와 사전을 비교하여 가장 긴 일치하는 구문을 찾는다. 일치하는 구문은 거리, 길이, 그리고 다음 문자 형태로 표현된다. 여기서 "거리"는 일치하는 구문의 시작점과 사전 내 슬라이딩 윈도우의 현재 위치 사이의 거리를 의미하며, "길이"는 일치하는 구문의 길이를 의미한다. "다음 문자"는 일치하는 구문을 따르는 문자열이다. LZ77이 각 데이터를 대체한 후, 허프만 코딩이 적용된다. 허프만 코딩은 데이터 내에서 심볼의 출현 빈도를 계산한다. 이후, 출현 빈도에 따라 허프만 트리를 구성한다. 심볼의 빈도가 높을수록 트리의 레벨이 높아진다. 트리 구성 후, 높은 빈도를 가진 심볼에는 짧은 비트를 할당하고, 낮은 빈도를 가진 심볼에는 긴 비트를 할당하여 데이터를 압축한다. 허프만 코드의 수식(1)은 다음과 같다.

$$L = 2 + \log_2(N) \quad (1)$$

여기서 L은 각 패턴에 할당된 비트 문자열의 길이를 나타내며, N은 전체 데이터에서 해당 패턴이 반복되는 횟수를 의미한다. 해당 수식을 통해 자주 사용되는 패턴은 먼저 처리되어 짧은 비트 문자열로 대체된다.

2.3 오토 인코더

오토 인코더는 반지도학습의 일종으로 데이터 압축에 효과적이라고 알려져있다. 오토 인코더는 크게 입력값을 압축하는 인코딩 모델과 인코딩된 값을 복구하는 디코딩 모델로 구성되어있다. 인코딩 모델의 입력값은 코드라고 불리는 중간 모델로 맵핑되게 된다. 인코딩 모델의 입력값을 $X = \{x_1, x_2, x_3 \dots x_n\}$라고 할 때 인코더는 다음의 수식(2)를 따른다.

$$Z = f(X) \quad (2)$$

이 때 코드 값 $Z = \{z_1, z_2, z_3 \dots z_n\}$는 압축된 데이터를 의미한다. 이후 디코더 모델은 Z를 입력으로 하여 원본 데이터 X와 유사하게 만들도록 학습하는 아래의 수식(3)을 따른다.

$$R = g(Z) \quad (3)$$

$R = \{r_1, r_2, r_3, \dots r_n\}$은 디코더에 의해 재구성된 출력 데이터이다. 학습 과정에서 오토 인코더는 입력 데이터와 재구성된 출력 데이터 사이의 차이를 최소화하도록 학습된다. 일반적으로 오토인코더는 손실함수로 평균제곱오차를 사용하고, 이를 최소화하도록 훈련한다. 오토인코더의 손실 함수 $Loss_{AE}$ 다음 수식(4)과 같다.

$$Loss_{AE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - r_i)^2. \quad (4)$$

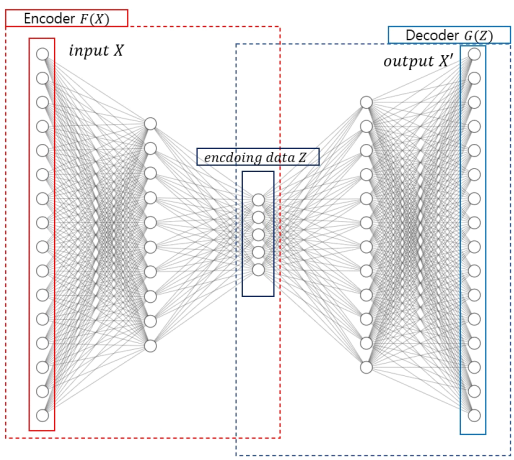

Fig. 1. Autoencoder Structure

III. 딥러닝 기반 부채널 데이터 압축 기법

3.1 기존 압축 방식의 한계

대표적인 무손실 압축 알고리즘인 Deflate 압축 알고리즘은 LZ77과 허프만 코딩을 조합한 압축 알고리즘이고 주로 gzip파일에 사용된다. 부채널 분석 분야의 많은 연구자은 Grizzly[48], ASCAD[49], DPAcontest V2 및 V4 데이터셋등을 zip과 gzip을 통해 압축하여 배포한다. Deflate 알고리즘은 데이터의 패턴이 자주 반복되고, 그 반복되는 패턴이 길수록 압축의 효율성이 증대된다. 주로 부채널 파형 수집을 위해 사용되는 오실로스코프는 샘플링된 전압 또는 전류값을 저장하는 장치로 입력되는 신호를 일정한 시간 간격으로 샘플링하여 디지털 값으로 변환한다. 이때 수집된 파형의 데이터는 8비트[50] 혹은 12비트[51] 데이터 포맷으로 표현되고, 이는 파형의 높낮이를 결정한다. 반복되는 연산이 많은 암호의 파형 수집 시, 데이터가 반복되는 패턴이 많이 있는 것처럼 보이지만, 같은 암호 연산이어도 입력에 따라서 중간값이 다를 경우 파형의 값이 미묘하게 달라진다. 반복되는 패턴이 다른 데이터에 비해, 8비트 혹은 12비트로 표현된 부채널 파형은 기존의 Deflate 방식으로는 효율적으로 압축하기 힘든 데이터의 형태를 가지고 있다. 또한, 파형의 값을 IEEE 754에 제시된 부동 소수점로 표현될 수 있다. IEEE 754는 컴퓨터에서 부동 소수점을 표현하기 위해 사용되는 표준으로써 최상위비트 순서대로 부호, 지수, 가수로 표현되는 방식이다. 가수부의 경우 실제 값의 소수점 이하 부분이 조금만 변경되어도 비트의 조합이 달라진다. 이러한 데이터 변동성 때문에 IEEE754 형식으로 기록된 부채널 분석 파형 역시 Deflate 방식으로는 효율적으로 압축하기 힘든 데이터 형태를 가지고 있다.

3.2 오토인코더 기반 압축 기법

본 논문에서는 그림 2와 같이 오토인코더를 이용한 부채널 데이터 압축 프레임 워크을 제시한다. 오토인코더는 반지도 학습의 일종으로 다양한 데이터[52]에서 압축의 뿐만 아니라 노이즈 감쇠[29]에 효과가 있다고 알려져있다. 부채널 데이터의 경우는 압축기가 단순히 데이터를 압축하는 것이 아닌 부채널적인 특성을 유지한 채 데이터를 압축할 수 있어야 하고, 압축을 해제하는 과정에서 부채널적인 특성이 사라지면 안된다. 부채널적인 특성이란 수집한 전력 혹은 전자파의 데이터가 암호 알고리즘이 동작할 때 아래의 수식(5)을 만족한다는 것이다.

$$Power = O + HW(Data) + Noise \tag{5}$$

O의 경우 고정된 오프셋이고 $HW(Data)$는 데이터의 헤밍 웨이트를 의미하고 $Noise$는 0을 평균으로 하고 σ를 표준편차로 하는 정규 분포를 따른다. 부채널 분석에 사용되는 데이터는 일반적으로 헤밍 웨이트 모델을 따른다고 가정하며, 데이터를 압축할 시 헤밍 웨이트를 기반으로 소비하는 전력의 정보나 경향성을 잃지 않아야 한다. 헤밍 웨이트의 정보를 바탕으로 압축되지 않는다면, 상관 전력 분석을 진행할 경우, 원본 파형을 통한 분석은 성공하나, 디코딩된 파형의 경우 분석이 되지 않을 수가 있다. 본 논문에서 제시하는 오토인코더 모델의 경우 부채널 데이터의 특성 손실되지 않을 뿐만 아니라 디코딩 속도을 기존 무손실 알고리즘과 동일하게 유지할 수 있었다. 인코딩 모델에서 부채널적인 특성을 유지한 채 데이터를 압축하기 위해 본 논문은 다음과 그림 2와 같은 압축기를 구성하였다.

제시한 모델의 인코딩 모델의 특징은 기존의 네트워크의 파라미터 개수나 연산량 자체를 줄이는 역할을 하는 풀링 레이어[54]를 제거했다는 것이다. 풀링 레이어의 경우 차원 축소에는 효과적이나 기존의 목적인 부채널적인 특성을 압축과정에서 손실 없이 유지하는 것이 불가능했다. 그러나 풀링 레이어를 제거하는 경우, 데이터의 크기가 커지기 때문에, 이를 보완하기 위해 모델 마지막에 완전 연결 계층를 추가하여 부채널 데이터를 최종적으로 압축했다. 첫번째 레이어에서는 부채널 특성을 각 포인트별로 정확하게 학습하기 위하여, 각각의 필터가 독립적으로 학습하는 로컬 연결 레이어[21]를 사용하였다. 컨볼루션널 네트워크[53] 같은 경우 모든 필터가 각 노드에 동일하게 적용되어 학습하는 구조이기 때문에 각각의

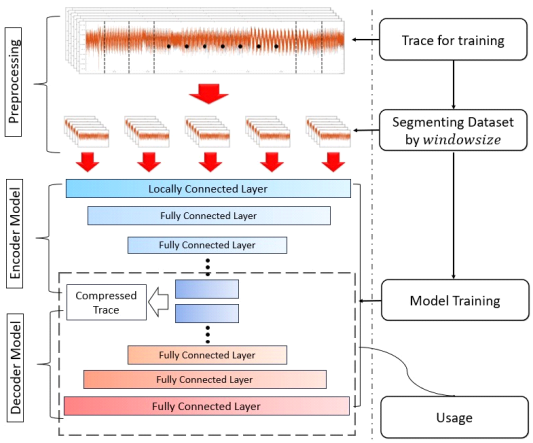

Fig. 2. Proposed Leakage Compression Structure based on Autoencoder

Training Autoencoder for Leakage Compression

Input: Trace set for training $T = \{T_1, T_2, T_3 \dots T_N\}$, length of the trace L, number of traces N
Output: Trained Model $M = \{M_1, M_2, M_3, \dots M_W\}$

1. $W \leftarrow \lceil L \,\%\, windowsize \rceil$ // Calculate the number of models
2. $Z \leftarrow \{Z_1, Z_2, Z_3 \dots Z_W\}$ // is the training datatset seperated by windowsize
3. for $i = 0$ to W do
4. $\quad idx_s \leftarrow i * windowsize$
5. $\quad idx_e \leftarrow idx_s + windowsize$
6. $\quad Z_i \leftarrow T[idx_s : idx_e]$
7. end for
8. Make Models $E = \{E_1, E_2, E_3, \dots E_W\}$ for compressing each Z_i
9. Make Models $D = \{D_1, D_2, D_3, \dots D_W\}$ for decompressing data
10. for $i = 0$ to W do
11. $\quad$ Train $M_i = D_i(E_i(Z_i))$
12. end for

Fig. 3. Algorithm of Training Autoencoder for Leakage Compression

암호의 연산의 단위를 정확하는 데 한계가 존재했다. 그러나 로컬 연결 레이어을 사용하면 각 필터가 스트라이드마다 다르게 학습되므로, 각각 암호의 연산의 특징을 정확하게 학습할 수 있었다. 본 모델에서 사용한 로컬 연결 레이어는 필터 크기 3, 스트라이드 1로 설정하였다. 디코더 모델의 경우, 인코딩된 부채널 데이터를 다시 복구하는 모델로, 실제로 사용자가 압축된 부채널 데이터를 다시 딥러닝 연산을 통하여 복구를 해야 하기 때문에 저장의 효율성과 본 데이터 복구를 위한 연산량을 고려하여 멀티 레이어 퍼셉트론[22]만을 이용하여 모델을 구성하였다.

오토 인코더를 학습하는 과정을 알고리즘화 하면 Fig. 3.과 같다. 부채널 데이터를 압축하기 위해서 각 파형은 특정 windowsize로 분할하고, 각 분할된 부채널 분석 데이터는 모델 M의 입력값으로 사용 되어 학습이 된다. 모델 M은 인코딩 모델 E와 D로 구성된다.

제안한 부채널 데이터 압축기를 평가하는 요소는 두 가지이다. 첫 번째로 제시한 압축기를 이용한 압축기는 Deflate 등 기존의 압축 알고리즘으로 압축한 파형보다 압축률이 높아야 한다. 이 때 TS_o를 압축하기 전 파형의 크기라고 하고 TS_C를 압축 후의 파형의 크기라고 하면 압축률은 다음 수식(5)와 같이 정의 할 수 있다.

$$CR = \frac{TS_o}{TS_c} \tag{6}$$

두 번째 요소는 부채널적인 특성을 유지하는 것이다. 본 논문에서는 압축기가 부채널적인 특성을 보존하는지 확인하기 위해서 압축 전과 후의 상관 전력 분석을 수행했다. 압축기 부채널적인 특성을 잘 보존한다면 두 종류의 파형을 대상으로 한 상관 전력 분석 결과가 동일해야 한다. 추가로 상관계수가 가장 높게 나타나는 위치 또한 동일 해야한다.

IV. 실험 결과

4.1 AES 부채널 파형 압축

본 절에서는 제시한 압축기를 통해 AES가 동작할 때 수집한 파형을 압축하고 압축한 파형을 대상으로 상관 전력 분석을 통해 부채널적 특성의 보존을 확인한다. 실험에 사용된 파형은 Chipwhisperer-lite[55] STM32F Board로 수집했으며, 사용된 파형은 부동 소수점형식으로 저장되어 있다. 각각의 파형은 5000개의 시점을 가지며 총 1.1G의 15000개의 파형을 압축하였다. 표 1은 windowsize별 부채널 데이터의 압축률과 압축 및 압축 해제 시간을 나타낸다. T_C는 파형을 압축하는 시간이고, T_D는 파형을 다시 압축해제하는 시간이다. Eff는 제안된 압축기의 압축 효율성이 Deflate와 비교했을 때 크기적으로 얼마나 효율적인지 나타내는 지표이다. 제시한 모델은 Deflate와 비교하여 압축 해제 시간이 2초로 유사하면서도, 압축

Table 1. Comparison of Compression Results for Each Compression Ratio

windowsize	*CR*	T_C	T_D	TS_C	*Eff*	Correlation Coefficient
Deflate	5.7	221	2	231	1	0.89
1000	10	12.2	3.4	114	1.7	0.91
	20	28.78	13.14	68	3.5	0.9
	40	27.04	3.06	28	7.01	0.9
	80	28.64	3.08	14.3	14.02	0.9
	200	26.8	3.08	5.72	35.04	0.89
2500	10	48.81	5.91	114	1.70	0.89
	20	55.13	4.69	57.8	3.50	0.89
	24	54.68	4.2	45.7	4.38	0.89
	50	32.32	2.11	22.8	8.76	0.88
	100	31.71	2.11	11.4	17.5	0.67
5000	2	32.7	5.91	286	1.70	0.81
	10	31	2.46	114	3.50	0.76
	20	31.19	1.85	57.2	7.01	0.73
	40	31.29	1.75	28.6	14.02	0.68
	100	32.1	1.62	11.4	35.04	0.57

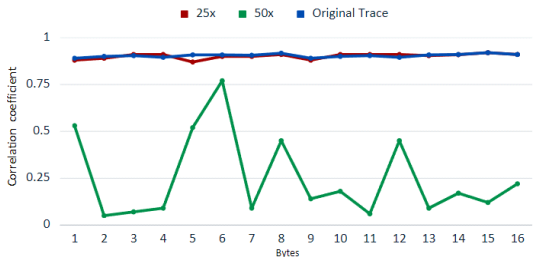

Fig. 4. Correlation Coefficient Values for Each Block

시간은 18배 빠르고 유사한 상관계수를 유지할 수 있었다. 또한 Deflate는 압축 후 파형의 크기가 231MB이지만 오토인코더를 사용한 압축기는 최대 11.4MB으로 기존의 압축 방식보다 효율적임을 입증하였다.

압축 모델이 부채널 데이터에 적합하다면 압축 비율에서 효율적이어야 하고, 원본 부채널 데이터의 특성을 유지해야한다. Fig.4.는 windowsize가 2500일 때 AES의 중간 값인 $Sbox\,[plaintext \oplus key]$을 이용해 압축 후 복구된 파형을 대상으로 상관 전력 분석을 시도한 결과이다. 상관 전력 분석 결과 16바이트 모두 원본과 상관계수가 유사했다. 그러나 압축 비율이 과도하게 높은 경우, 예를 들어 압축 비율이 100배인 경우, 기존의 상관계수 0.89에 비해 낮은 0.67을 기록했다. 100배의 압축한 파형 역시 비밀키를 찾는 데 충분한 상관계수를 가지지만, 원본 파형에 비해서 상관계수가 낮기 때문에 정확하게 복구했다고 보기 어렵다.

Fig. 5.은 파형의 개수를 증가시키면서 각 키에 대한 최대 상관계수를 보여준다. 파형의 개수를 증가시키면서 각 키에 대한 최대 상관계수를 나타낸 결과에서는 성공적인 압축/압축 해제의 경우, 올바른 키에 대해 원본 파형과 매우 유사한 결과를 확인할 수 있었다. 성공적인 압축/압축해제의 경우 Fig. 5b, 5c, 5d에서 결과는 올바른 키에 대해 원본 파형과 동일하거나 유사한 상관계수 패턴을 확인할 수 있다. 또한 제안된 압축기의 경우 오토인코더에 잡음 감소 특성을 따른다. Fig. 5a.의 원본 파형이 일부 잘못된 키에 대해 0.3에서 0.4 사이의 상관 계수를 기록했다. 그러나 Fig. 5b, 5c, 5d.와 같은 경우 옳지

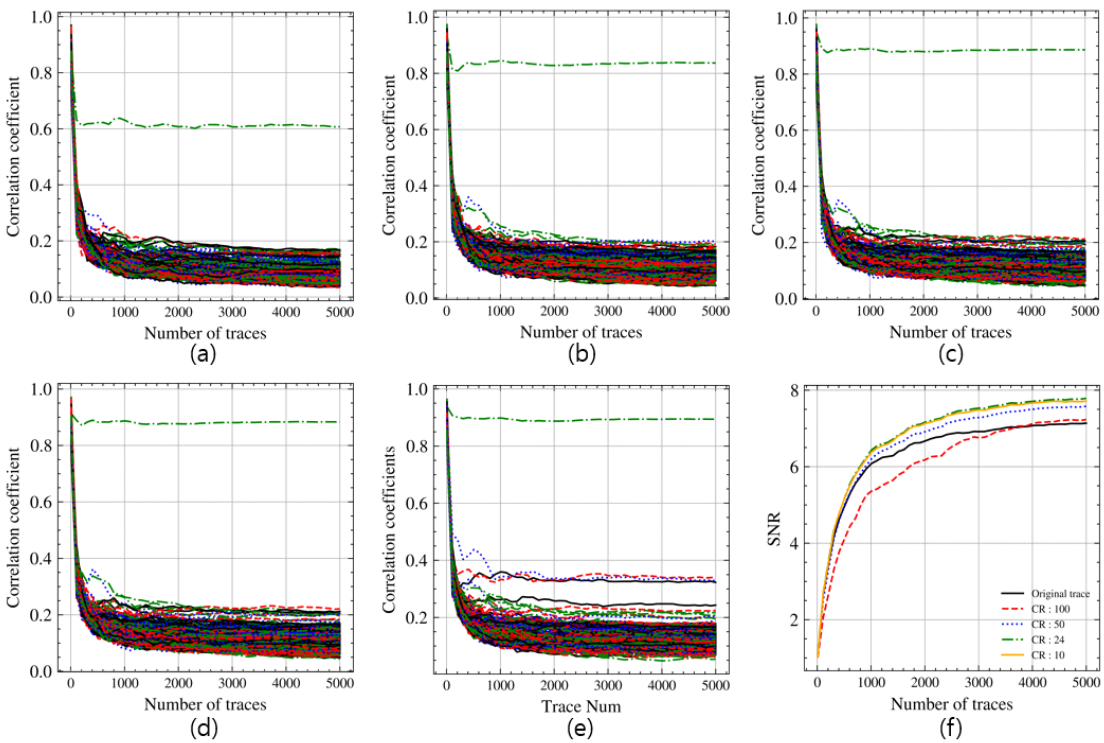

Fig. 5. CPA Results per CR After Decompressing Traces

않은 키를 추측할 때의 상관계수가 0.2로 수렴하여 올바른 키를 쉽게 구분할 수 있었다. 압축 비율이 너무 높아지는 경우, Fig. 5e.의 경우처럼, 키 복구는 가능하지만 원본과 동일한 상관 계수 값을 얻는 것은 불가능했다. Fig. 4f.는 신호 대 잡음 비율(SNR)은 원본 파형와 비교할 때 압축이 부채널 특성을 보존했다면, 모든 구간에서 제시한 압축기를 거친 파형들이 SNR 값이 높았다.

window size가 2500일 때 압축 비율에 따른 최대 피크의 위치를 보여주는 Table. 2,에서는, 상관계수를 유지하는 압축 모델의 경우 원본 파형에서 최대 피크의 위치가 동일하게 유지되었음을 확인할 수 있다. 하지만 더 높은 압축 비율에서는 디코딩 실패로 인해 대부분의 바이트에서 원본 파형과 다른 위치에서 피크가 관측되었다.

Table 2. Comparing Points with the Highest Correlation Coefficients in the Correlation Power Analysis Results of Intermediate Values of ith Bytes.The ith byte Corresponds to $Sbox[plaintext[i] \oplus key[i]]$

Method / *ibytes*	Raw Data	Proposed Method		
	1	10	50	100
1	1190	1190	1190	1190
2	1230	1230	1230	4516
3	1270	1270	1270	2738
4	1310	1310	1310	839
5	1350	1350	1350	1355
6	1390	1390	1390	1653
7	1430	1430	1430	3213
8	1470	1470	1470	2660
9	1510	1510	1510	2705
10	1550	1550	1550	1545
11	1590	1590	1590	2059
12	1630	1630	1630	4716
13	1670	1670	1670	3045
14	1710	1710	1710	2770
15	1750	1750	1750	1812
16	1790	1790	1790	836
Success Rate(%)	100	100	100	7

4.2 마스킹 AES 부채널 파형 압축

부채널 분석 대응 기술 중 하나인 마스킹은 암호 연산 중에 발생하는 중간값을 랜덤한 값을 이용하여 숨기는 방법이다. 본 절에서 제시한 압축기 또한 마스킹이 적용된 파형에 대해서 압축하고 해제했을 때, 마스킹에 대한 특성 또한 보존해야한다. 모델이 대응 기술이 적용된 부채널 파형에도 부채널 적인 특성을 유지한 채 압축이 가능한지 실험하기 위해 1차 마스킹 기법이 적용된 ASCAD[49] 데이터를 이용하여 압축을 시도 하였다. ASCAD 데이터는 8bit ATMmega8515에서 1차 마스킹이 적용된 AES가 동작할 때 수집한 전자파 정보이다. 파형의 하나 시점은 8bit로 구성되어 있고 100000개의 시점이 존재한다. 해당 파형은 각 시점의 값이 [-40, 60]에 존재한다. 각 노드의 값들의 편차가 증가함에 따라서, 모델의 성능 감소를 고려하여 ASCAD 압축 모델을 학습할 때는 훈련동안 파형의 평균값을 제거했다.

표 3은 ASCAD를 windowsize 10000으로 압축한 결과이다. 원본 파형은 5.59G이고 해당 파형을 Deflate 알고리즘을 통해 압축하면 2.76G이다. 제시한 압축기의 경우 파형이 압축 해제 시간의 효율성이 기존의 Deflate 알고리즘보다 최대 약 8배 효율적이다. 또한 Deflate 알고리즘은 원본 데이터를 2배 압축하였지만, 논문에서 제안하는 모델의 경우 부채널적인 특성을 보존하는 최대 압축시 143MB까지 파형을 압축할 수 있었고, 이는 기존의 Deflate 방식보다 약 46배 효율적이었다.

압축기가 마스킹 값의 부채널적인 특성을 보존하기 위해, 총 두가지 상관 전력 분석 결과를 제시한다. 첫 번째로 마스킹 값을 알고 있다는 가정하에 상관 전력 분석을 시행한다. 이 때 1차 상관 전력 분

Table 3. Comparision of Compression Results for ASCAD

	Proposed Method				Deflate
CR	10	20	40	100	2
T_D	73.5	65.7	62.6	60.3	520
T_S	572	286	143	57.3	2760
Eff	4.94	9.89	19.7	49.4	·
Correlation Coefficient	0.21	0.21	0.21	0.08	0.21

석을 위해 사용되는 중간값은 $Sbox[plaintext \oplus key] \oplus masking$이다. Fig. 6.은 원본 파형과 복구한 파형의 1차 상관 전력 분석 결과이다. 최대 상관계수는 원본 파형이 0.92, 복구한 파형이 0.93으로 소폭 상승한 결과를 확인할 수 있다. 이 결과 마스킹 값이 포함된 Subtyes의 결과값을 성공적으로 분석할 수 있었다.

두 번째는 마스킹 값을 알고 있지 않다는 가정하에 2차 상관 전력 분석을 시행했다. 이 때 사용한 중간값은 마스킹 값을 알고 있지 않다는 가정하에 $Sbox[plaintext \oplus key]$으로 설정했다. Fig. 7.은 이차 상관전력 분석의 결과이다. 100000 포인트 파형의 [45400, 46100] 구간을 추출하여, 2차 상관전력 분석을 시행했다. Fig. 7(a).은 복구한 파형, Fig. 7(b).는 원본 파형에 2차 상관 전력 분석 결과

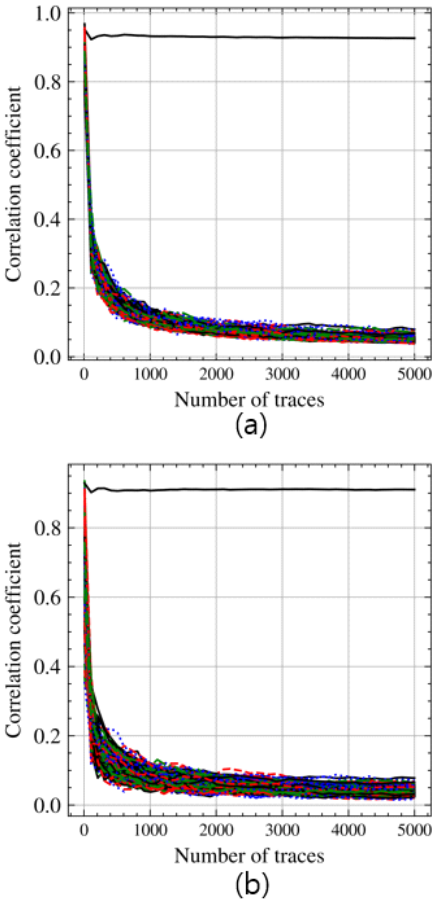

Fig. 6. First-order CPA Results with a Known Masking Value (a): Our work, (b): Original

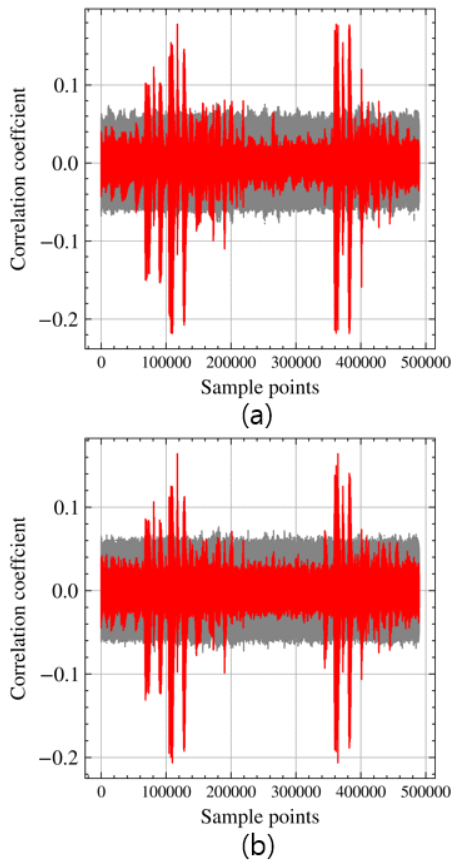

Fig. 7. Second-order CPA Results (a): Our work, (b): Original

이다. 두 가지 결과 모두 옳은 키를 맞게 분리한 것을 확인할 수 있고, 가장 큰 피크의 위치와 값이 모두 동일한 것을 보아 마스킹이 적용된 암호에서도 본 논문이 제시한 압축기를 사용할 수 있다는 것을 확인할 수 있다.

V. 결 론

데이터 크기의 폭발적인 증가로 인해, 효율적인 압축 알고리즘이 중요해졌다. 본 논문은 딥러닝을 사용하여 채널 데이터를 효율적으로 압축하는 새로운 접근법을 소개한다. 이 연구에서는, 인코더가 로컬 연결된 층을 활용하는 오토인코더 기반 데이터 압축기를 사용한다. 이를 통해 채널 신호 내의 개별 점들을 학습할 수 있게 하여, 각 점에 대한 채널 특유의 특성을 획득할 수 있다. 디코더는 간단한 MLP 모델로 설계되어 Deflate 방법에 비해 빠른 압축 해제

를 보장한다. 제안된 압축기는 Deflate 알고리즘에 비해 최대 8배 더 효율적인 압축을 달성하며, 압축 및 압축 해제 과정에서 대부분의 부채널 특성을 보존한다. 고샘플링률이 요구되는 환경에서 데이터 크기를 줄임으로써, 저장 요구사항, 전송 대역폭 및 처리 오버헤드를 감소시킬 수 있는 큰 이점을 제공할 수 있다.

References

[1] Deutsch and Peter. DEFLATE compressed data format specification version 1.3. No. rfc1951. 1996.

[2] Deutsch and Peter. GZIP file format specification version 4.3. No. rfc1952. 1996.

[3] Burtscher, Martin, and Paruj Ratanaworabhan. "FPC: A high-speed compressor for double-precision floating-point data," IEEE transactions on computers vol58.1 pp. 18-31, Jan 2009.

[4] Di, Sheng, and Franck Cappello. "Fast error-bounded lossy HPC data compression with SZ," 2016 ieee international parallel and distributed processing symposium (ipdps). IEEE, pp. 730-739, May 2016.

[5] Lindstrom and Peter. "Fixed-rate compressed floating-point arrays," IEEE transactions on visualization and computer graphics, vol 20.12, pp 2674-2683, No.v 2014.

[6] Lakshminarasimhan, Sriram, and et al. "Compressing the incompressible with ISABELA: In-situ reduction of spatio-temporal data," Euro-Par 2011 Parallel Processing: 17th International Conference, Euro-Par 2011, Bordeaux, France, August 29-September 2, 2011, Proceedings, Part I 17, pp. 366-379, Sep. 2011

[7] Wallace and Gregory K. "The JPEG still picture compression standard," Communications of the ACM vol 34.4, no. 4, pp. 30-44, Feb. 1992.

[8] Le Gall and Didier. "MPEG: A video compression standard for multimedia applications," Communications of the ACM vol. 34.4, pp. 253-262,

[9] Dávid Sztahó, Kiss Gábor, and Tulics Miklós Gábriel. "Deep learning solution for pathological voice detection using LSTM-based autoencoder hybrid with multi-task learning," I14th International Joint Conference on Biomedical Engineering Systems and Technologies, pp. 135-141, Jan. 2021.

[10] Otter, Daniel W., Julian and et al. "A survey of the usages of deep learning for natural language processing," IEEE transactions on neural networks and learning systems 32.2, pp. 604-624, Oct 2020.

[11] Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks," science 313.5786, pp. 504-507, Jul 2006.

[12] Pawar and Aashay. "Noise reduction in images using autoencoders," 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS). IEEE, pp. 987-990, Dec. 2020.

[13] Mao, Xiao-Jiao, and et.al "Image restoration using convolutional auto-encoders with symmetric skip connections," arXiv preprint arXiv:1606.08921, Jun 2016.

[14] Tien, Chin-Wei, et al. "Using autoencoders for anomaly detection and transfer learning in IoT," Computers 10.7, pp. 88, July 2021.

[15] Liu, Tong, and et al. "High-ratio lossy compression: Exploring the autoencoder to compress scientific

data," IEEE Transactions on Big Data 9.1 ,pp. 22-36, Mar. 2021

[16] Sento and Adna. "Image compression with auto-encoder algorithm using deep neural network (DNN)," 2016 Management and Innovation Technology International Conference (MITicon). IEEE, pp. 99-133, Oct. 2016.

[17] Momenifar, Mohammadreza, and et al. "A physics-informed vector quantized autoencoder for data compression of turbulent flow," 2022 Data Compression Conference (DCC). IEEE,pp. 1-10, Jan. 2022.

[18] Lou, Xiaoxuan, et al. "A survey of microarchitectural side-channel vulnerabilities, attacks, and defenses in cryptography," ACM Computing Surveys (CSUR) 54.6 , pp. 1-37, Jan. 2022

[19] Khan, Haider Adnan, and et al. "IDEA: Intrusion detection through electromagnetic-signal analysis for critical embedded and cyber-physical systems," IEEE Transactions on Dependable and Secure Computing 18.3, pp. 1150-1163, Aug. 2019

[20] Sehatbakhsh, Nader and et al. "REMOTE: Robust external malware detection framework by using electromagnetic signals," IEEE Transactions on Computers 69.3, pp. 312-326, Oct. 2019.

[21] Chen, Yu-hsin, et al. "Locally-connected and convolutional neural networks for small footprint speaker recognition," Sixteenth Annual Conference of the International Speech Communication Association, Sep. 2015

[22] Driss, S. Ben, et al. "A comparison study between MLP and convolutional neural network models for character recognition." Real-Time Image and Video Processing 2017. Vol. 10223. SPIE, pp. 32-42, May. 2017.

[23] Maghrebi, Houssem, Thibault Portigliatti, and Emmanuel Prouff. "Breaking cryptographic implementations using deep learning techniques," Security, Privacy, and Applied Cryptography Engineering: 6th International Conference, SPACE 2016, Hyderabad, India, Proceedings 6. Springer International Publishing, pp. 3-26, Dec. 2016.

[24] Cagli, Eleonora, Cécile Dumas, and Emmanuel Prouff. "Convolutional neural networks with data augmentation against jitter-based countermeasures: Profiling attacks without pre-processing." Cryptographic Hardware and Embedded Systems - CHES 2017: 19th International Conference, Taipei, Taiwan, Proceedings, Springer International Publishing, pp.45-68, Sep. 2017.

[25] Kim, Jaehun, and et al. "Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis," IACR Transactions on Cryptographic Hardware and Embedded Systems, pp. 148-179, May. 2019.

[26] Das, Debayan, and et al. "X-DeepSCA: Cross-device deep learning side channel attack," Proceedings of the 56th Annual Design Automation Conference 2019. pp. 1-6, Jun. 2019

[27] Timon and Benjamin. "Non-profiled deep learning-based side-channel attacks with sensitivity analysis," IACR Transactions on Cryptographic Hardware and Embedded Systems, pp. 107-131, Feb. 2019.

[28] Kwon, Donggeun, Seokhie Hong, and Heeseok Kim. "Optimizing implementations of non-profiled deep learning-based side-channel attacks," IEEE Access, pp. 5957-5967, Jan. 2022.

[29] Kwon, Donggeun, Heeseok Kim, and Seokhie Hong. "Non-profiled deep learning-based side-channel preprocessing with autoencoders," IEEE Access 9 , pp. 57692-57703. Apr 2021

[30] Wold, Svante, Kim Esbensen, and Paul Geladi. "Principal component analysis," Chemometrics and intelligent laboratory systems 2.1-3 , pp. 37-52. Aug, 1987

[31] Standaert, François-Xavier, and Cédric Archambeau. "Using subspace-based template attacks to compare and combine power and electromagnetic information leakages," International Workshop on Cryptographic Hardware and Embedded Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 421-425, Aug. 2008.

[32] Wang, Ping, and et al. "Enhancing the performance of practical profiling side-channel attacks using conditional generative adversarial networks," arXiv preprint arXiv:2007.05285, Jul 2020.

[33] Goodfellow, Ian, and et al. "Generative adversarial nets," Advances in neural information processing systems 27, Jun. 2014.

[34] Zhou, Yuanyuan, and François-Xavier Standaert. "Deep learning mitigates but does not annihilate the need of aligned traces and a generalized resnet model for side-channel attacks," Journal of Cryptographic Engineering 10.1, pp. 85-95, Apr. 2020

[35] Paguada, Servio, Lejla Batina, and Igor Armendariz. "Toward practical autoencoder-based side-channel analysis evaluations," Computer Networks 196, Sep. 2021

[36] Ramezanpour, Keyvan, Paul Ampadu, and William Diehl. "SCAUL: Power side-channel analysis with unsupervised learning," IEEE Transactions on Computers 69.11, pp. 1626-1638, Jan. 2020

[37] Perin, Guilherme, Lichao Wu, and Stjepan Picek. "Exploring feature selection scenarios for deep learning-based side-channel analysis." IACR Transactions on Cryptographic Hardware and Embedded Systems 2022.4, pp. 828-861, Aug. 2022

[38] Li Mu, and Wangmeng Zuo, and et al. "Learning convolutional networks for content-weighted image compression," Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3214-3223, June 2018.

[39] Liu Tong, Jinzhen Wang, and et al. "High-ratio lossy compression: Exploring the autoencoder to compress scientific data," IEEE Transactions on Big Data 9.1, pp. 22-36, Mar 2021.

[40] Alexandre, David, and et al. "An autoencoder-based learned image compressor: Description of challenge proposal by NCTU," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 2539-2542, Feb. 2019.

[41] Cheng, Zhengxue, and et al. "Energy compaction-based image compression using convolutional autoencoder," IEEE Transactions on Multimedia 22.4, pp. 860-873, Apr. 2020.

[42] Fournier, Quentin, and Daniel Aloise. "Empirical comparison between autoencoders and traditional dimensionality reduction methods," 2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). IEEE, pp. 211-214, Jun. 2019

[43] Lempel, Abraham, and Jacob Ziv. "On the complexity of finite sequences," IEEE Transactions on information theory 22.1, pp. 75-81, Jan. 1976

[44] Huffman, David A. "A method for the construction of minimum-redundancy codes," Proceedings of the IRE 40.9 , pp. 1098-1101, Sep. 1952

[45] Kocher, Paul, Joshua Jaffe, and Benjamin Jun. "Differential power analysism," Advances in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, Proceedings 19. Springer Berlin Heidelberg, Aug. 1999.

[46] Brier, Eric, Christophe Clavier, and Francis Olivier. "Correlation power analysis with a leakage model," Cryptographic Hardware and Embedded Systems-CHES 2004: 6th International Workshop Cambridge, MA, USA, Proceedings 6. Springer Berlin Heidelberg, pp. 16-29, Aug. 2004.

[47] Marcellin, Michael W., and et al. "An overview of JPEG-2000," Proceedings DCC 2000. Data compression conference. IEEE, pp. 3-48, Jan, 2002

[48] Choudary, Omar. "MGK: Grizzly: power-analysis traces for an 8-bit load instruction," 2017.

[49] Benadjila, Ryad, et al. "Deep learning for side-channel analysis and introduction to ASCAD database," Journal of Cryptographic Engineering 10.2, pp. 163-188, Jun. 2020.

[50] Goller, Gabriel, and Georg Sigl. "Side channel attacks on smartphones and embedded devices using standard radio equipment," International Workshop on Constructive Side-Channel Analysis and Secure Design. Cham: Springer International Publishing, pp. 255-270, Apr. 2015.

[51] Krupa, M., and M. Gasior. "Precise Digital Integration of Fast Analogue Signals Using a 12-bit Oscilloscope," Proceedings of 3rd International Beam Instrumentation Conference (IBIC 2014). pp. 584-586, Sep. 2014.

[52] Graa, Mariem, and et al. "Detection of side channel attacks based on data tainting in android systems," ICT Systems Security and Privacy Protection: 32nd IFIP TC 11 International Conference, SEC 2017, Rome, Italy, Proceedings 32. Springer International Publishing, pp. 205-218, May. 2017

[53] O'shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint arXiv:1511.08458, Nov. 2015.

[54] Ranzato, Marc'Aurelio, Y-Lan Boureau, and Yann Cun. "Sparse feature learning for deep belief networks," Advances in neural information processing systems 20. Jan. 2008.

[55] O'flynn, Colin, and Zhizhang Chen. "Chipwhisperer: An open-source platform for hardware embedded security research," Constructive Side-Channel Analysis and Secure Design: 5th International Workshop, COSADE 2014, Paris, France, Revised Selected Papers 5. Springer International Publishing, pp. 243-260, Apr. 2014.

〈저 자 소 개〉

정 상 윤 (Sangyun Jung) 학생회원
2023년 2월: 고려대학교 과학기술대학 인공지능사이버보안학과 학사
2023년 3월~현재: 고려대학교 일반대학원 사이버보안 석사과정
〈관심분야〉 부채널 공격

진 성 현 (Sunghyun Jin) 정회원
2015년 2월: 서울시립대학교 수학, 컴퓨터과학 학사
2017년 2월: 고려대학교 정보보호대학원 공학석사
2022년 8월: 고려대학교 정보보호대학원 공학박사
2022년 9월~현재: 삼성전자 메모리사업부 보안 소프트웨어 엔지니어
〈관심분야〉 정보보호, 부채널 공격, 머신러닝 기반 암호분석

김 희 석 (Heeseok Kim) 종신회원
2006년: 연세대학교수학과학사
2008년: 고려대학교정보보호대학원석사
2011년: 고려대학교정보보호대학원 박사
2011년 9월~2012년 12월: Bristol University 박사후연구원
2013년~2016년 8월: 한국과학기술정보연구원(KISTI) 선임연구원
2015년~2016년 8월: 과학기술연합대학원대학교(UST) 조교수
2016년 9월~현재: 고려대학교 과학기술대학 인공지능사이버보안학과 부교수
〈관심분야〉 부채널 공격, 암호시스템 안전성 분석 및 고속구현, 암호칩 설계 기술, 보안관제, 네트워크 보안