Neural distinguisher for PIPO

https://youtu.be/j9wv06Z6IsY

PIPO block cipher

Dataset, Model architecture, and Training

Evaluation

Conclusion

PIPO block cipher

- ICISC'20에서 발표된 경량 블록암호
- **64-bit 평문과 128-bit, 256-bit의 키를 지원**
- 키의 길이에 따라 라운드가 다름 (64/128 → 13-round, 64/256 → 17-round)
- 각 라운드에서는 S-layer (S-box), R-layer (rotation), Key addition을 반복
- **입력 차분 : 0x8000088008088000**

데이터 셋, 모델 구성 및 학습

Dataset

- 랜덤 평문 2개 선택 (두 평문은 차분 관계가 아님)
- 하나의 평문에 **입력 차분을 XOR**
- 총 3개의 평문을 암호화
- **랜덤 암호문 쌍은 0**, **차분을 갖는 평문에 대한 암호문 쌍은 1**로 라벨링

Algorithm 1. Dataset preparation

Input: Input difference (δ), The number of data (N_{ds}), Encryption function (ENC)

Output: Dataset (DS)

$\delta = 0x8800088008088000$

for i=0 **to** $N_{ds}/2$ **do**

Choose random plaintext $P_0, P_1 (P_1 \neq P_0 \oplus \delta)$

$P'_0 = P_0 \oplus \delta$

$C_0 = ENC(P_0)$

$C_1 = ENC(P_1)$

$C'_0 = ENC(P'_0)$

$C_0 \| C_1$ is labeled 0 (Random)

$C_0 \| C'_0$ is labeled 1 (Cipher)

$DS \leftarrow C_0 \| C_1$ and $C_0 \| C'_0$

end for

return DS

Model architecture for neural distinguisher for PIPO

- 사용한 모델 구조는 다음과 같음
- **MLP 모델 사용**
- 2라운드 데이터가 1라운드에 비해 복잡한 데이터이므로 **모델 크기를 늘림**
  - 1라운드 차이임에도 불구하고, **19개의 레이어 추가 + 각 레이어의 유닛의 수도 2배**로 증가시킴
- 이진분류에 맞는 손실함수 (BCELoss)와 일반적으로 성능이 좋은 Adam 최적화함수 사용
- 약간의 과적합이 발생 → 해결하기 위해 **Dropout layer**추가 + 전체 구조를 **Residual network**로 구성

Rounds	1	2
Architecture (Unit of hidden layer)	MLP with 1-layer (units = 128)	MLP (Residual) with 20-layers (units = 256)
Batch size	32	8
Activation	ReLu	
Optimizer (Learning rate)	Adam (lr=0.001)	
Loss function	BCELoss	
Dropout	0.5	

Training

- 다른 작업들과 마찬가지로 입력-은닉-출력층을 순서대로 통과
- Loss와 accuracy 구한 후, epoch 반복
- **최종 accuracy가 0.5보다**
  - 크다면 : 랜덤 암호문 쌍과 차분을 갖는 암호문 쌍을 구별 가능한 모델
  - 작다면 : 구별 불가능한 모델
- 따라서, **$acc > 0.5$ 인 경우에만 neural distinguisher로 사용**

Algorithm 2. Training process

Input: Dataset (DS), Input layer (I), Hidden layer (H), Output layer (O), The number of hidden layer (h), The number of epoch ($EPOCH$)
Output: Trained model (M)
for epoch=1 **to** $EPOCH$ **do**
  $x \leftarrow I(\text{Data of DS})$
  for i=0 **to** h **do**
    $x \leftarrow H(x)$
  end for
  $out \leftarrow O(x)$
  $loss \leftarrow BCELoss(out, \text{Label of DS})$
  Compute accuracy (acc)
  Update parameters of neural network model
end for
If $acc \leq 0.5$ **then**
  Abort M
else if $acc > 0.5$ **then**
  return M

실험 및 평가

실험 환경

- **Apple M1 Pro 16GB RAM, Python 3.8.9, Pytorch 1.12.0**
- **메모리 문제**로 인해 다수의 학습 데이터를 사용하는 것이 불가능
  - **1,2 라운드** 암호문 데이터에 대한 실험만 진행
- GPU 51GB RAM을 제공하는 Google Colaboratory Pro Plus (클라우드 플랫폼) 사용 시도
  - 메모리 문제로 인해 800만개 이상의 데이터에 대한 학습이 불가능
- 이로 인해 **2-round에 대한 결과까지만 확인**

평가

- **1, 2라운드에 대해 높은 정확도** (각각 1.0, 0.98)로 암호문과 랜덤 데이터 구별 성공
- 정확도가 0.5보다 큰 경우 신경망 구별자로 사용 가능
  - 따라서, 해당 모델은 **2라운드 데이터에 대해 0.48 (=0.98-0.5) 의 신뢰성을 갖는 신경망 구별자로 사용 가능**
- 1라운드 증가 시 파라미터의 수가 **80배 증가**
- **3라운드의 경우 800만개의 데이터 사용 → 그러나 분류 실패** (그 이상은 메모리 문제로 실험에 실패)
- 라운드가 늘어날수록 필요한 데이터의 수, 파라미터의 수가 **큰 폭으로 늘어날 것으로 예상**

Round		1	2
The number of parameters		16879	1349121
Epoch		20	
The number of data	Training	$2^{20.7}$	
	Validation	$2^{20.2}$	
	Test	$2^{16.46}$	
Accuracy	Training	1.00	0.98
	Validation	0.99	0.98
	Test	1.00	0.98

결론 및 향후 연구

- **결론**
  - 블록암호 **PIPO에 대한 차분분석을 위한 딥러닝 기반의 신경망 구별자**를 제안
  - 실험 환경의 제약으로 인해 1, 2 라운드에 대해서만 실험
  - 각각 **1.00, 0.98의 높은 정확도를 달성** **(2라운드에 대해 0.48의 신뢰성 확보)**
  - **제안 기법이 PIPO 블록암호의 차분 분석을 위해 신뢰성이 높은 신경망 구별자로 사용 가능함을 보임**

- **향후 연구**
  - 더 많은 라운드에 대한 실험을 위해 **메모리 문제를 해결하기 위한 방법 찾기**
  - 또 다른 인공지능 기술 또는 전처리 과정을 통해 **효율적인 신경망 구별자를 설계**

감사합니다.