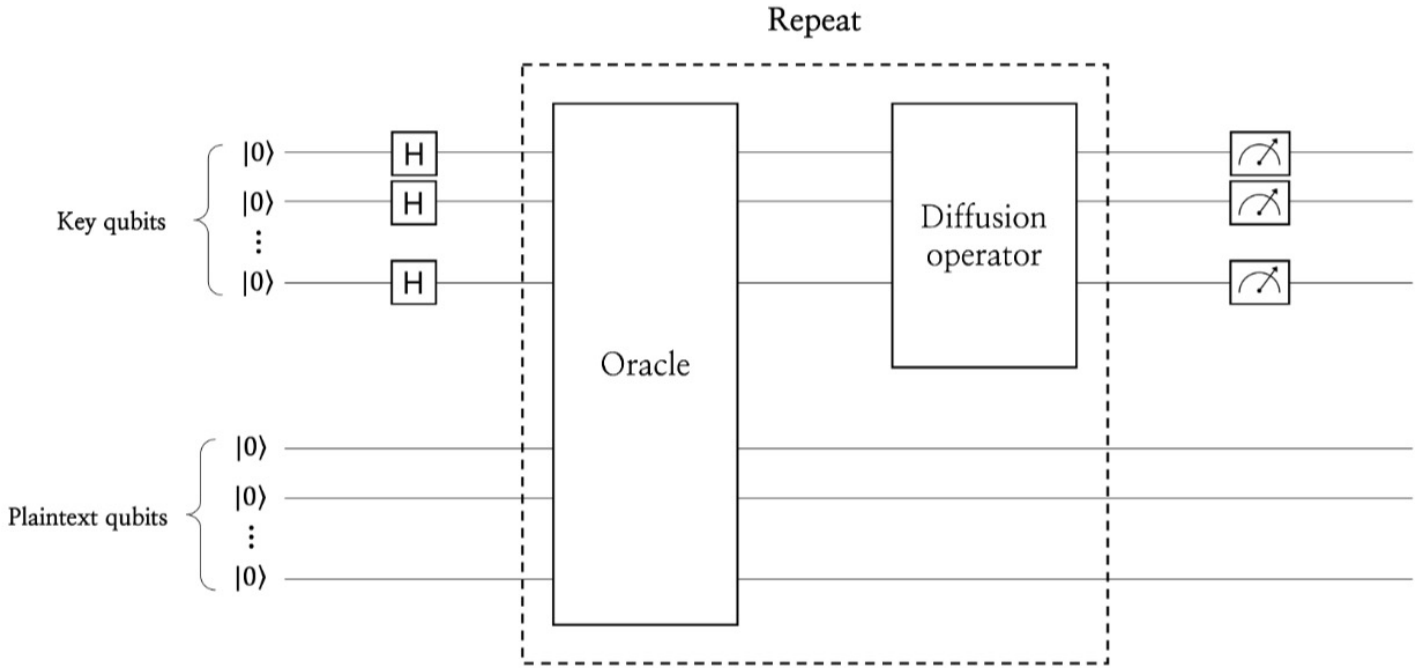
암호연구회 1차 발표

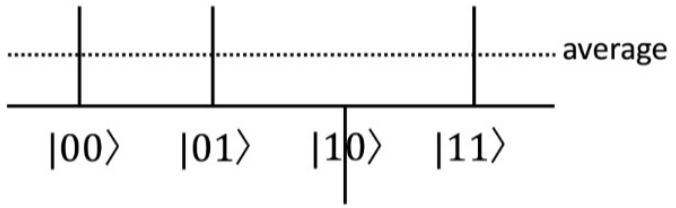
https://youtu.be/BQg-Y4oszTU

Grover algorithm

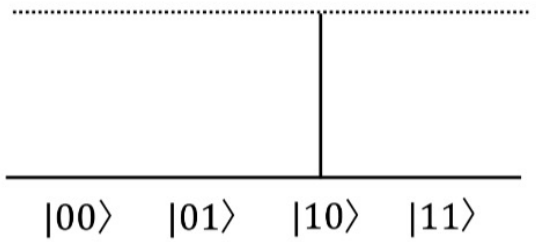
- 중첩 상태의 key를 이용하여 대칭키 암호에 대하여 brute-force attack을 수행하는 알고리즘
  - **Oracle :** 주어진 평문-암호문 쌍에 대한 키를 반환, 공격 대상의 암호화가 양자 회로로 구현되어야 함
  - **Diffusion operator :** Oracle에서 반환한 키의 진폭을 증폭시켜 관측 확률 증가
- 하지만 현재 양자컴퓨터의 성능 한계(qubit 수, 오류 등)로 실제 양자컴퓨터로 동작은 불가능
- 양자 컴퓨터의 가용 자원이(ex. 사용 가능한 qubit 수) 암호 공격에 필요한 자원에 도달할 때가 곧 암호가 깨질 수 있는 시점으로 예상



<Grover key search 회로 구조>



<Oracle>



<Diffusion operator>

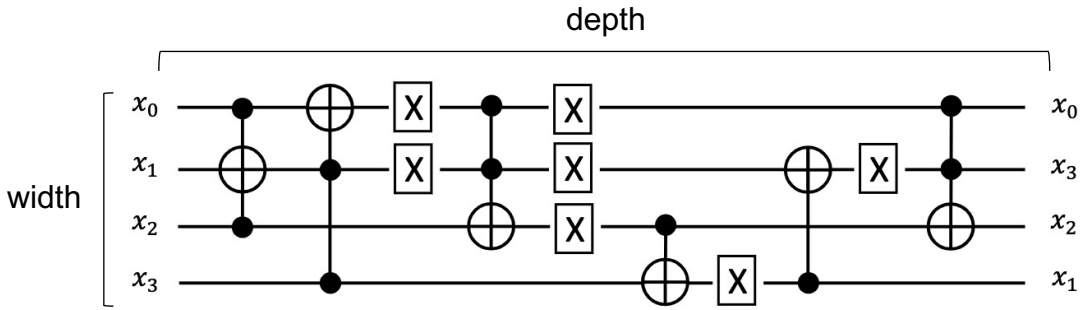
그루버 알고리즘에 필요한 양자 자원 추정

- **Grover algorithm 사용시 굉장히 많은 양자 자원 필요**

→ 실제 양자 컴퓨터로 Grover attack 가능 시기가 불분명 (자원 달성 뿐만 아니라 양질의 자원 및 오류 정정 필요)

목적 알고리즘	양자 자원					
DES-56	Qubit	CCCNOT	CCNOT (Toffoli)	CNOT	X	Depth
	6648	-	1.36×2^{39}	1.54×2^{40}	1.44×2^{40}	1.22×2^{39}
AES-128	Qubit	CCCNOT	CCNOT (Toffoli)	CNOT	X	Depth
	984	-	1.96×2^{77}	1.02×2^{78}	1.07×2^{71}	1.21×2^{77}
S-AES	Qubit	CCCNOT	CCNOT (Toffoli)	CNOT	X	Depth
	32	1.57×2^{11}	1.17×2^{14}	1.76×2^{14}	1.71×2^{12}	1.15×2^{15}

Quantum circuit



- **Width(너비)** : 물리적인 큐비트 수

- **Depth(깊이)** :
  - 입력에서 출력까지의 경로에 대한 최대 길이
  - 연속적인 **양자 게이트 연산의 수**를 의미하며 물리적 구현 시간을 나타냄 (즉, 연속적인 게이트 수가 depth로 이어짐)
  - **Depth가 클수록 양자 컴퓨터의 noise가 커지므로 원하는 결과를 얻기 어려움**

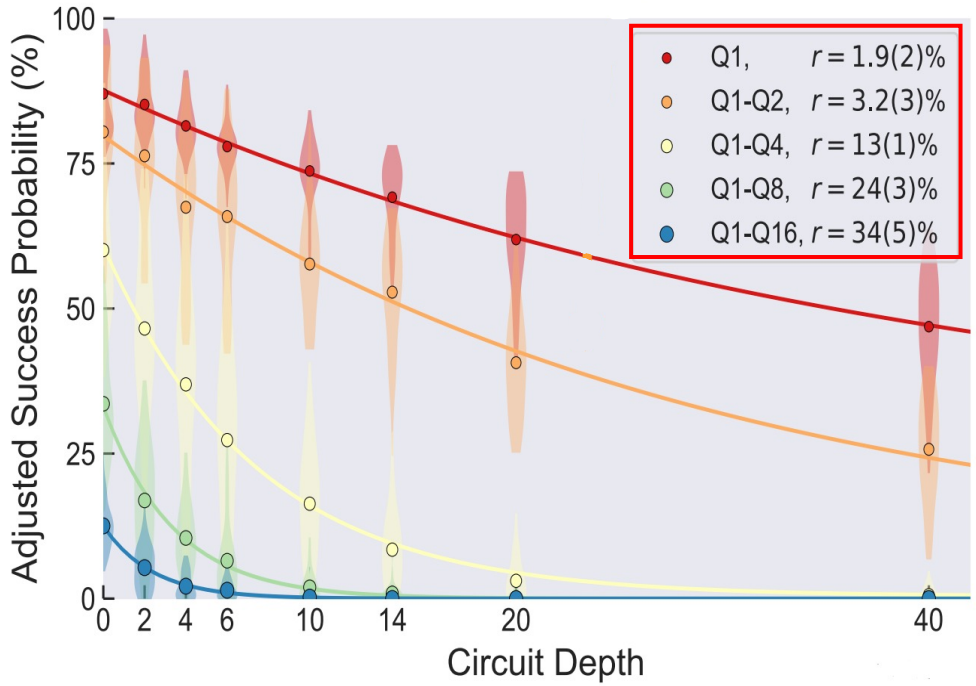
- 현재 얕은 depth의 알고리즘은 실제 양자 컴퓨터에서 실현 가능
- 물리적인 큐비트 수를 달성하더라도, 큰 depth의 양자회로의 경우 오류로 인해 동작이 어려움
  → 물리적 큐비트 수 달성 뿐만 아니라 Depth 최적화가 중요함 (ex. 사용 양자 게이트 수 줄이기, 병렬 구현 등)

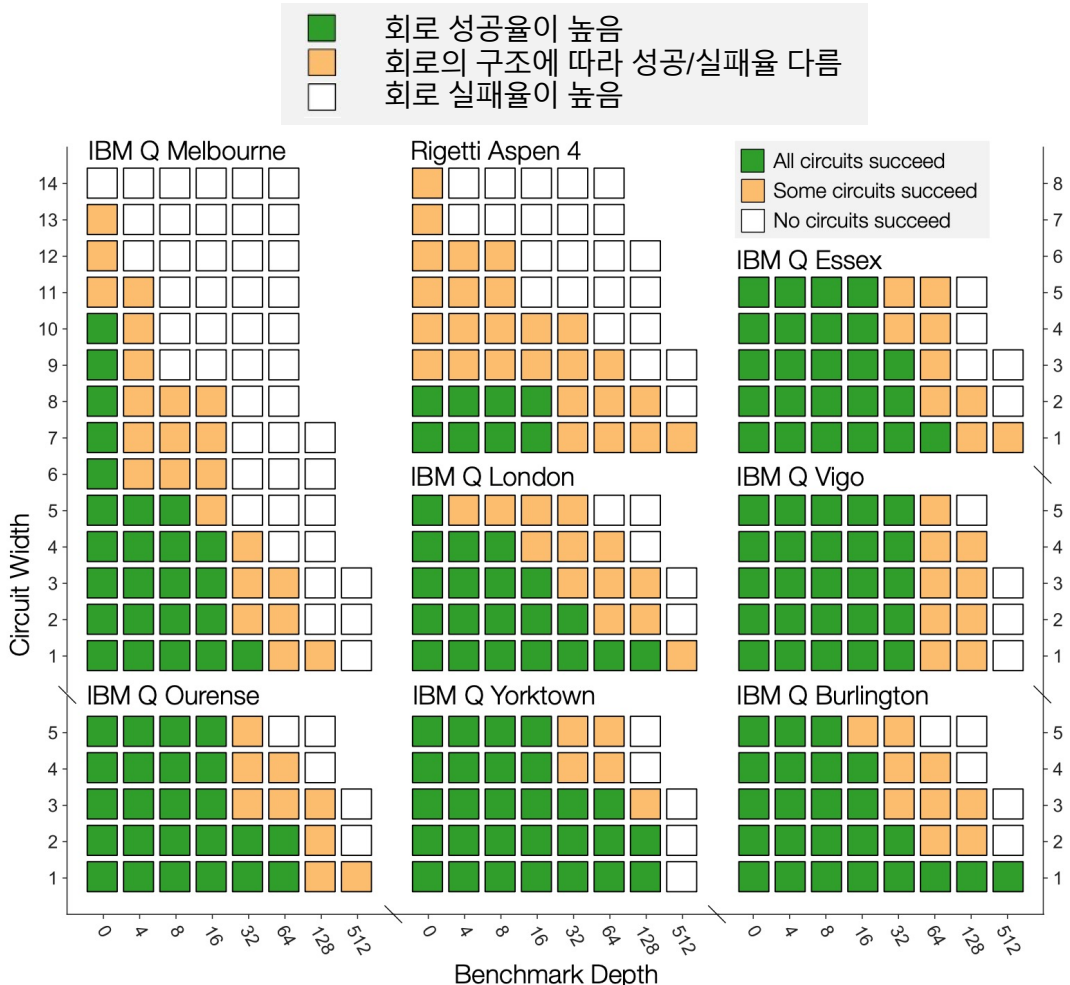
- Noise 가 많은 양자 컴퓨터를 사용하기 위한 방안 :
  1. **양자 오류 수정** : 오류 수정을 위한 많은 오버헤드 비용 필요
     - 추가적인 큐비트, 게이트가 많이 필요하므로 양자컴퓨터를 사용하기 위한 시간이 훨씬 오래 걸림
  2. **Depth 최적화 :** 양자 게이트 수를 줄인 최적화 구현 및 회로의 병렬 구현 등

Quantum circuit

- **Width, Depth에 따른 양자회로 동작 벤치마킹 결과** (IBM 양자컴퓨터 기준)
  - Width, Depth가 커질수록 회로의 오류율 증가 (성공률 감소)
  - 양자 회로 동작을 위해서는 Width(큐비트 수), Depth(연속적인 양자 게이트 연산 수) 모두 중요

Width(큐비트 수)
Ex) Q1-Q4 : 4큐비트 양자회로





Quantum circuit

- 양자컴퓨터 실행을 위해 필요한 2가지

1. 물리적인 양질의 큐비트 자원 수 달성

- 큐비트 수 뿐만 아니라 양자 게이트(제어된 얽힘 동작)을 정확도 있게 수행할 수 있는 양질의 큐비트가 중요함
- 최상의 하드웨어의 2 큐비트 게이트 당 오류율은 0.1% 이상 (일반적으로는 더 큼)

2. 양자컴퓨터 noise 제어

- **회로의 게이트 수 및 depth를 줄여 오류율을 줄임**

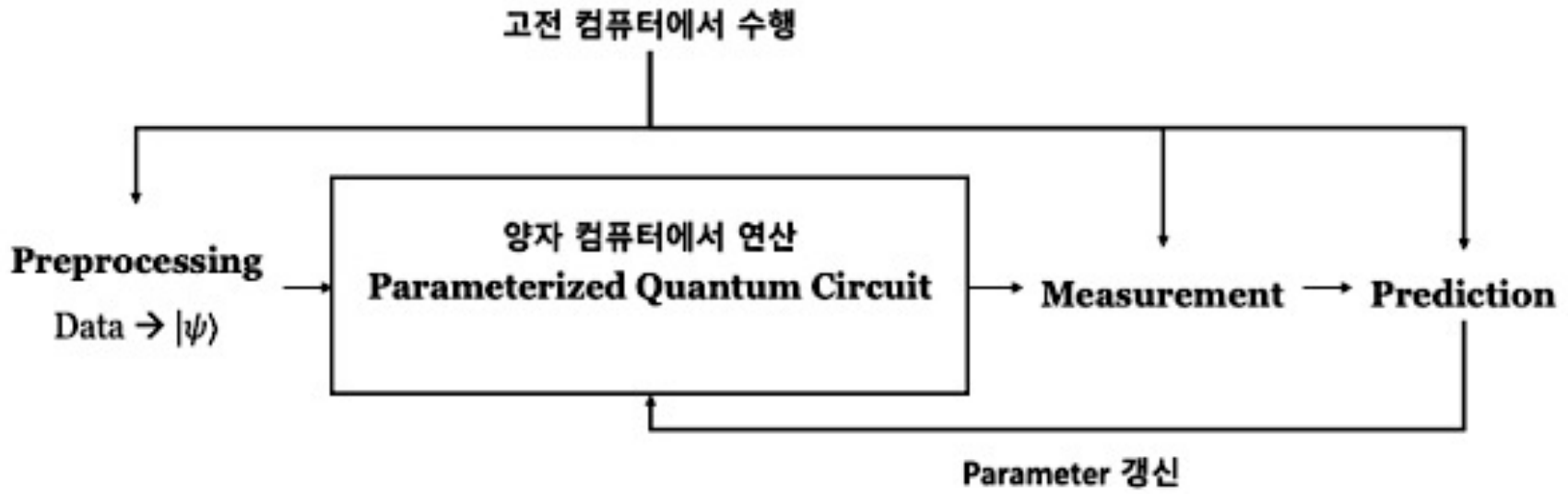
→ 가장 이상적이고 많이 연구되는 방향

- **오류수정을 통한 양자회로 확장** (양자 게이트, 큐비트에 굉장히 많은 오버헤드 비용이 필요)

→ 양자 오류 수정을 통한 회로 확장은 아직은 먼 목표

Quantum AI

- 양자 신경망은 **전통적인 신경망의 학습 과정을 양자 회로로 구성**한 것
- 전통적인 신경망과 같이 **확률론적 공격 수행 가능**
- 고전 컴퓨터상의 데이터를 양자컴퓨터 상에서 연산하기 위해 **임의의 큐비트 상태로 인코딩**한 후, **양자회로를 동작** 시킴
- 큐비트에 오류가 있어도 정확도 손실은 있으나 **일정 수준 이상의 정확도를 확보할 경우 공격 가능**



<Quantum Neural Network 구조>

한성대 연구팀 구현 결과물 소개 – Quantum AI

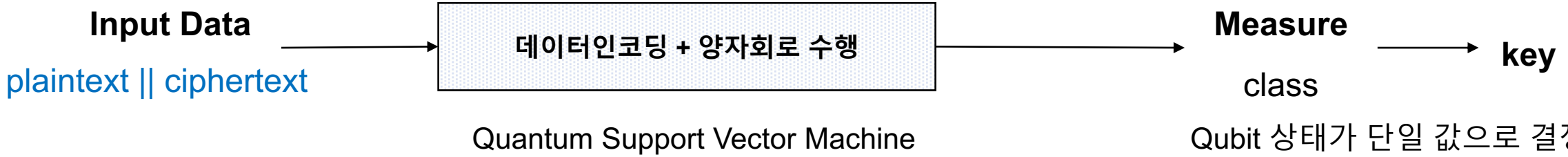
알고리즘	논문	발표년도
Caesar	Kim et. al. Cryptanalysis of Caesar using Quantum Support Vector Machine. IEEE ICCE-ASIA'21.	2021

한성대 연구팀 구현 결과물

- **시뮬레이터와 실제 양자하드웨어 상에서 QSVM을 통해 Caesar 암호에 대한 known-plaintext attack 수행**
- **Known-plaintext attack**
  → 공격자가 평문과 암호문에 접근할 수 있을 때, 비밀키를 찾는 공격
- **기존 데이터를 양자 상태로 변환하는 인코딩 과정을 거쳐 양자 회로로 표현**
- **양자 회로의 반복실행을 통해 훈련과 분류가 가능**

<데이터셋 형태>

Plaintext bit		Ciphertext bit		Key
0	1	0	0	3
⋮	⋮	⋮	⋮	⋮
1	1	0	1	2
Data				Label



한성대 연구팀 구현 결과물 소개 – Quantum AI

- **실행 시간은 사용된 큐비트 수에 따라 다름**(큐비트 수가 늘어날수록 실행 시간이 늘어나거나 실행이 완료되지 않았음)
- Float형 / Bit형 데이터셋으로 타입을 나누어 실험 수행 ➔ **Bit형 데이터셋에서 높은 분류 정확도 달성**
  - Float형은 큐비트의 수가 적어 실행시간이 비교적 적게 들지만, 데이터에 대한 정보가 적어 성능이 낮음
  - Bit형은 높은 데이터 차원을 가지므로 더 많은 큐비트가 필요하여 실행시간이 오래 걸리지만 성능이 높음
- **실제 양자 하드웨어는 'ibmq_bogota' 사용**
- **시뮬레이터보다 5.5배 이상의 시간 (780초) 소요 + 정확도 0.07 감소**
- **결론 : 양자 머신러닝 모델인 QSVM을 통해 Caesar 암호에 대한 암호 분석 가능(2, 3-bit key), 실제 양자하드웨어를 사용할 경우 분류 확률이 감소**
  → **실제 양자 하드웨어의 연산 시 발생하는 노이즈 등의 오류로 인해 정확도 손실**

Shots	2-bit dataset	3-bit dataset
1	0.66	0.6
5	1.0	0.7
100	-	0.81
150	-	0.84

양자 시뮬레이터 상에서의 shot에 따른 정확도
(2비트 데이터셋 : 4큐빗, 3비트 데이터셋 : 6큐빗)

			The number of shots	
			5	150
Bit type	The number of qubits	4	142.21	144.94
		6	1618.06	1867.55
Float type		2	110.27	114.09

양자 시뮬레이터에서
큐비트와 shots에 따른 수행시간 (unit : s)

Execution time	Accuracy
780	0.93

실제 양자 하드웨어 상에서의 정확도와 수행시간
(unit : s / the number of shots : 5 (best case),
2비트 데이터셋 : 4큐빗)

Quantum AI에 필요한 양자 자원 추정

목적 알고리즘	양자 자원
S-DES	2차 발표에 수행 예정
S-AES	
..	

Grover vs Quantum AI

- 양자 컴퓨터를 블록암호에 대한 공격 방법으로 양자 알고리즘인 Grover algorithm의 사용과 고전 AI에 양자의 성질을 사용하는 방법이 연구되고 있음

양자 자원이 적은 Quantum AI 가 더 빠르게 실현될 가능성이 높음
양자 자원을 달성하더라도 큐비트 오류의 정정이 필수적인 **Grover의 사용 시기는 더 늦을 것으로 예상**

Grover	Quantum AI
• **모든 key에 대한 brute-force attack 이므로 확실한 공격 (전수조사)** • **많은 양질의 양자 자원이 필요** • **큐비트에 오류가 있을 시 공격이 어려움**	• **암호의 패턴을 분석하는 확률론적 공격 (취약한 패턴 분석)** • **적은 큐비트 수와 작은 양자 회로 필요** • **큐비트에 오류가 있을 시 정확도 손실은 있으나 일정 수준 이상의 정확도를 확보할 경우 공격 가능**

Classical AI vs Quantum AI

- 고전 신경망과 함께 사용하는 hybrid quantum AI와 양자 회로만을 사용하는 Quantum AI가 존재
- Qiskit, Amazon Braket, Q# 등의 프레임워크에서 Quantum AI를 위한 기술 개발 중

현재 사용 가능한 큐비트가 많지 않으며, **Quantum AI에서는 더 적은 큐비트만을 활용**할 수 있기 때문에 **Hybrid Quantum AI를 사용하는 것이 더 적절**할 것

Classical AI

- **현재 다양한 분야에서 실제로 활용 되고 있으며, 다양한 라이브러리 및 학습 기술 존재**
- **학습 가속을 위한 하드웨어도 개발되었으며, 큐비트 등과 같이 제한적인 자원이 필요하지 않음**

Quantum AI

- **Classical AI와 함께 사용 가능 (기존의 딥러닝 관련 함수들과 더 적은 큐비트 사용 가능)**
- **그러나 학습을 위해 회로를 반복수행해야 하므로 많은 큐비트는 사용할 수 없음**
- **Quantum AI는 Classical AI 보다 고차원의 데이터를 다루는 데에 이점이 있음**
- **Classical RNN, LSTM이 약 2만개의 파라미터를 필요로 할 때, Quantum AI는 약 2천 개의 파라미터 필요[QRNN] (더 적은 파라미터 사용)**

[QRNN] Bausch, Johannes. "Recurrent quantum neural networks." Advances in neural information processing systems 33 (2020): 1368-1379.

분석 대상 암호 및 연구 동향

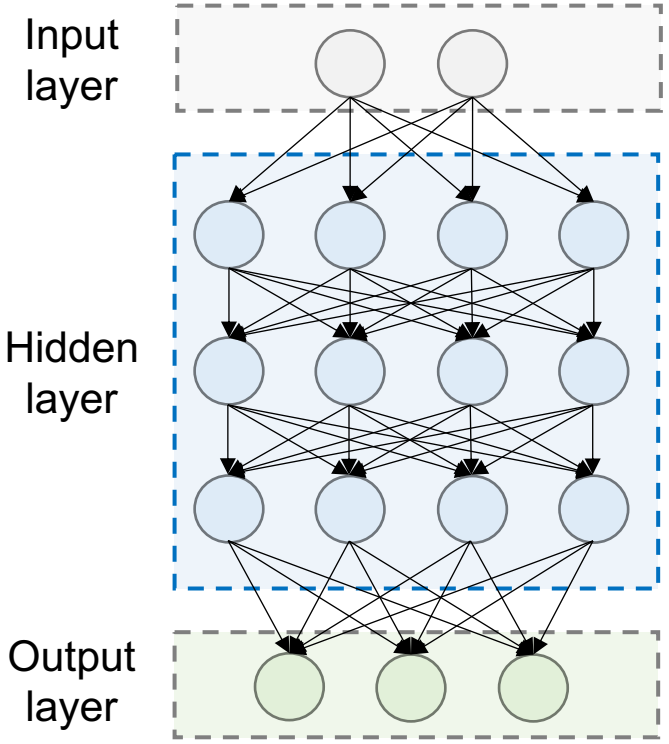
<table>
<tr><th>Grover algorithm</th><th colspan="2">Classical Neural Networks</th><th>Quantum Neural Networks</th></tr>
<tr><td rowspan="2">Caesar[Caesar&Vigenere], Vigenere[Caesar&Vigenere], DES[DES], S-AES[S-AES], GIFT[PRESENT, GIFT] , PRESENT[PRESENT, GIFT] , PIPO[PIPO], 등</td><td>알려진 평문 공격</td><td>차분 공격</td><td rowspan="2">Caesar [qsvm]</td></tr>
<tr><td>S-DES, Round-reduced SPECK and SIMON [So]</td><td>Round-reduced SPECK [Gohr], SIMON[simon], PRESENT[present], GIMLI, ASCON, KNOT, CHASKEY(모두 round-reduced)) [baksi]</td></tr>
<tr><td>양자 자원의 부족 및 오류로 인해 실제 공격 X</td><td>[So]는 텍스트 기반 키에 대해 공격 성공
(S-DES를 제외하면,랜덤 비트 키에 대해서는 실패)</td><td>Full-round에 대한 공격 X</td><td>QSVM을 통한 Caesar 암호 알려진 평문 공격 성공
(2-bit, 3-bit key)</td></tr>
</table>

[Caesar&Vigenere] Song, Gyeongju, et al. "Grover on Caesar and Vigenère Ciphers." Cryptology ePrint Archive (2021).
[DES]Jang, Kyung-bae, et al. "Quantum Cryptanalysis for DES Through Attack Cost Estimation of Grover's Algorithm." Journal of the Korea Institute of Information Security & Cryptology 31.6 (2021): 1149-1156.
[S-AES] Jang, Kyung-Bae, et al. "Grover on Simplified AES." 2021 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia). IEEE, 2021.
[PRESENT, GIFT] Jang, Kyungbae, et al. "Efficient implementation of present and gift on quantum computers." Applied Sciences 11.11 (2021): 4776.
[PIPO] Jang, Kyungbae, et al. "Grover on PIPO." Electronics 10.10 (2021): 1194.
[LEA] Jang, Kyung Bae, et al. "그루버 알고리즘 적용을 위한 LEA 양자 회로 최적화." 정보처리학회논문지/컴퓨터 및 통신 시스템 제 10.4 (2021): 4.
[So] So, Jaewoo. "Deep learning-based cryptanalysis of lightweight block ciphers." Security and Communication Networks 2020 (2020).
[qsvm] Kim, Hyun-Ji, et al. "Cryptanalysis of Caesar using Quantum Support Vector Machine." 2021 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia). IEEE, 2021.
[baksi] A. Baksi, J. Breier, X. Dong, and C. Yi, "Machine learning assisted differential distinguishers for lightweight ciphers." IACR Cryptol. ePrint Arch., vol. 2020, p. 571, 2020.
[simon] Hou, Zezhou, Jiongjiong Ren, and Shaozhen Chen. "Cryptanalysis of round-reduced simon32 based on deep learning." Cryptology ePrint Archive (2021).
[present] Jain, Aayush, Varun Kohli, and Girish Mishra. "Deep learning based differential distinguisher for lightweight cipher PRESENT." Cryptology ePrint Archive (2020).

인공 신경망 종류

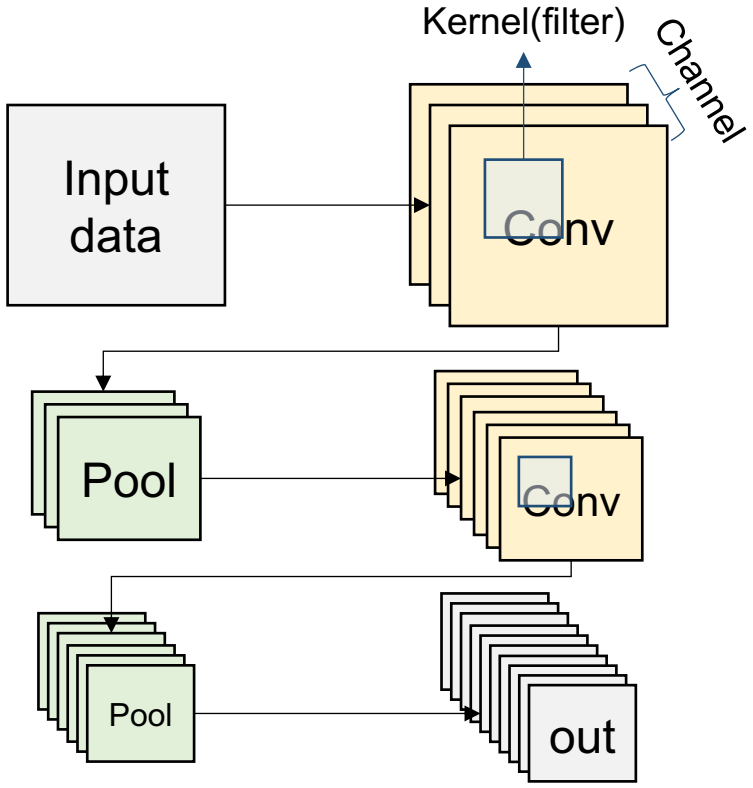
Linear Multi Layer Perceptron

- 신경망의 **가장 기본적인 형태**
- 2개 이상의 hidden layer 필요
- 모든 unit이 연결 되어 있음
- 시계열, 이미지 신경망보다는 전체적인 데이터 특징 고려



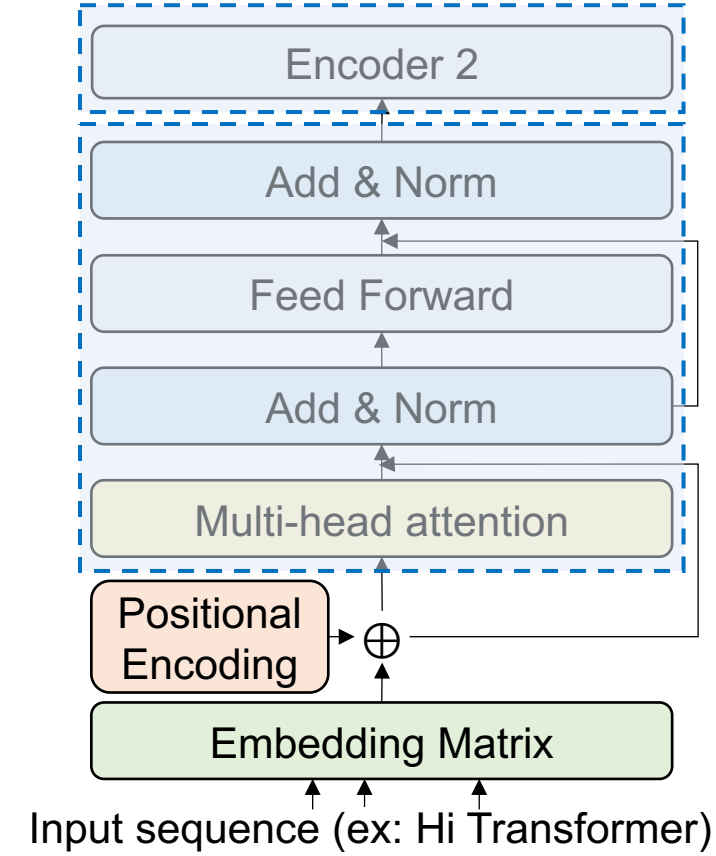
Convolutional Neural Networks

- **시계열(1d), 이미지(2d)에 효과적**인 신경망
- Conv + Pool 구조 & kernel(가중치 행렬)과 입력 데이터의 컨볼루션 연산 → **지역적 특징 학습**
- 출력(featuremap)의 크기는 줄어들며 특징 추출 및 강화, channel을 늘려 손실된 정보 보완



Transformer Encoder

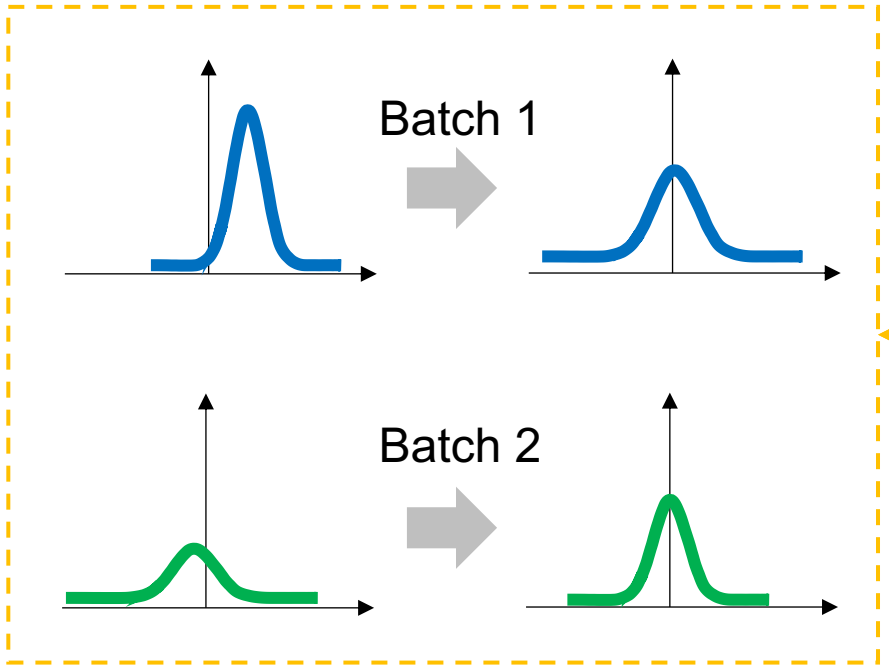
- 기존 **시계열** 신경망(RNN 등)의 단점을 보완한 신경망 (기계 번역 등에 활용)
- **Encoder + Decoder** 구조
- 입력 데이터가 한번에 입력
  → 순차적 구조 X, 병렬처리 용이



인공 신경망 구조

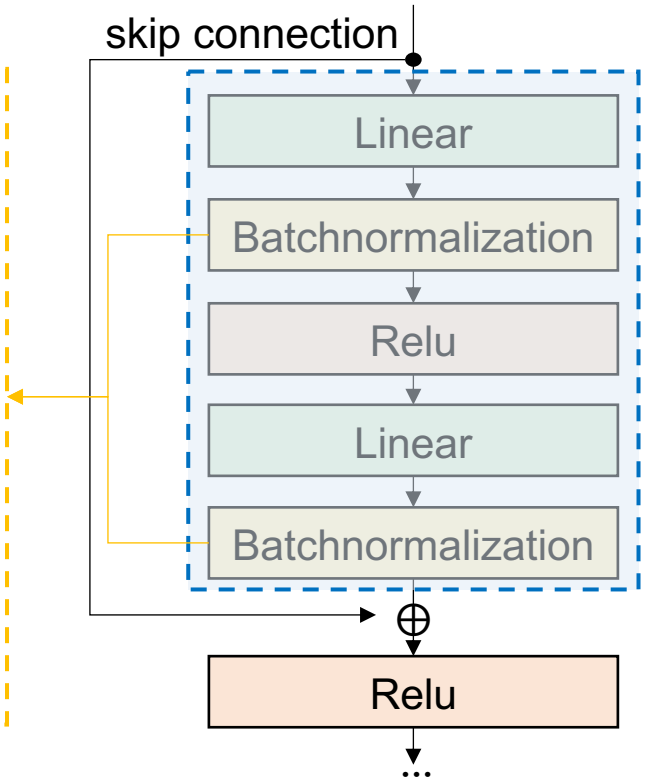
Batch normalization

- 학습 과정에서 평균과 분산을 조정
  → **정규 분포로 만들어 학습 안정화**
- 기울기 소실 방지 → 높은 학습률 사용 가능
  → **학습 가속화**



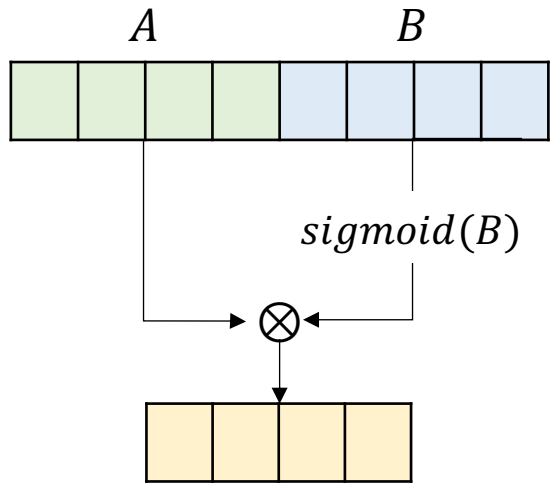
Residual Network

- Output을 몇 개의 layer를 건너뛴 후 input에 추가하여 **추가 정보 학습 가능**
  → **기울기 소실 방지, 과적합 방지**



Gated Linear Unit

- **정보의 출입을 제어**
  중요 정보 : sigmoid 곱에서 살아 남음
  아닌 경우 : 사라짐
  → **중요 요소에 집중 가능**
- **더 빠르고 안정적인 학습 가능**



고전 신경망을 통한 암호 분석

- **Data**
  - Data : plaintext || ciphertext
  - Label : key
- **Data type**
  - Bitstring (0 or 1)
  - Hexadecimal (0x0 ~ 0xF)
- **Neural Networks**
  - Linear Neural Network (MLP) (for bitstring)
  - 1-dimension Convolutional Neural Network (for bitstring)
  - Transformer Encoder + MLP (for hexadecimal)
- **Architecture**
  - Default
  - Batch normalization + Residual network (skip connection)
  - Batch normalization + Residual network (skip connection) + Gated Linear Units
- **Target cipher**
  - S-DES (10-bit key, 8-bit plaintext, 8-bit ciphertext)
  - S-AES (16-bit key, 16-bit plaintext, 16-bit ciphertext)

고전 신경망을 통한 암호 분석

- 암호화는 다음과 같은 성질을 가짐

  - **단일 비트가 변경**될 경우 **대부분 혹은 모든 비트에 영향** [혼돈]

  - **평문 1비트를 변경**할 경우 통계학적으로 **암호문의 절반이 변경** [확산]

- 예를 들어 평문의 1번째 비트가 전체 암호문에 영향을 줄 수 있기 때문에 사용할 데이터는 temporal locality를 갖기 어려우며, **전역적인 정보를 반영**해야 함

  → temporal locality를 갖는 데이터의 학습에 효과적인 Convolution 및 Recurrent NN 계열이 아닌 **전역적인 정보를 고려하기 좋은 linear layer 기반의 MLP가 적절**할 것으로 판단

*Temporal : time 정보 가짐
*Locality : 인접 feature는 비슷한 정보를 가짐

[혼돈] C. E. Shannon, "Communication theory of secrecy systems," in *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656-715, Oct. 1949, doi: 10.1002/j.1538-7305.1949.tb00928.x.
[확산] Stallings, William (2014). Cryptography and Network Security (6th ed.). Upper Saddle River, N.J.: Prentice Hall. pp. 67–68. ISBN 978-0133354690.

S-DES 암호 분석

- MLP (**10-bit key**, 8-bit plaintext and ciphertext)

MLP		1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th
Previous Work	BAP	0.64	0.74	0.71	0.58	0.64	0.8	0.54	0.6	0.85	0.8
	Dev	0.14	0.24	0.21	0.08	0.14	0.3	0.04	0.1	0.35	0.3
Residual	BAP	0.72	0.77	0.75	0.6	0.76	0.8	0.59	0.68	0.85	0.83
	Dev	0.22	0.27	0.25	0.1	0.26	0.3	0.09	0.18	0.35	0.33
Gated Linear Units	BAP	0.72	0.79	0.77	0.62	0.75	0.81	0.59	0.66	0.87	0.85
	Dev	0.22	0.29	0.27	0.12	0.25	0.31	0.09	0.16	0.37	0.35

BAP가 낮음 → 안전

BAP가 높음 → 취약

- 모든 비트의 Dev가 0이상이어야 공격 성공한 것으로 간주 [So]
- **이전 연구(69%)보다 약 5% 향상된 정확도(74%)**

*Bit Accuracy Probability (BAP) : 각 비트에 대한 예측 정확도
*Deviation (Dev) : BAP – 키 발생 확률

*Google Colaboratory (cloud service), Nvidia GPU (Tesla T4, 50GB RAM)
*학습 소요 시간 : 약 5~8시간

S-DES 암호 분석

- CNN (**9-bit key**, 8-bit plaintext and ciphertext)

CNN		1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th
Default + Gated Linear Units	BAP	1.0	0.59	0.51	0.52	0.50	0.81	0.53	0.51	0.84	0.73
	Dev	0.0	0.09	0.01	0.02	0.0	0.31	0.03	0.01	0.34	0.23

BAP가 높음

- **9-bit 까지 모든 비트에 대한 공격이 가능**
  - MLP의 10-bit key와 비슷하게 **6번째, 9번째, 10번째 비트는 취약**
  - 그러나 **높은 확률로 예측하는 경우는 MLP에 비해 적음**
- **10-bit**는 모든 비트에 대해 약 0.5 또는 0.49 → **학습이 되지 않음 & 공격 실패 (0.5 미만)**

S-DES 암호 분석

- Transformer encoder (**6-bit key**, 8-bit plaintext and ciphertext)

Transformer Encoder + MLP		1st	2nd	3rd
Default	HAP	1.0	0.73	0.53
	Dev	0.0	0.23	0.03
Gated Linear Units	HAP	1.0	0.73	0.54
	Dev	0.0	0.23	0.04

- Bit가 아닌 hexadecimal 데이터 (4-bit당 1 hexadecimal)
  → 총 12-bit로 표현된 10-bit key
- 6-bit key이므로 상위 6-bit가 0
  → 2번째는 경우의 수 적어서 73%를 달성한 것이고
  10-bit key일 때는 3번째와 비슷한 수준의 정확도 예상
- 시계열 신경망 + hexadecimal을 사용
  → **Bit type의 MLP 보다 적은 키 공간에 대한 공격만 가능**

*Hexadecimal Accuracy Probability (HAP) : 각 hexadecimal에 대한 정확도

S-DES 암호 분석

Network		MLP (10-bit key)			CNN (9-bit key)	Transformer Encoder (6-bit key)	
Architecture		Previous Work	Residual	GLU (Best case)	Default + GLU	Default	GLU
Layers (units)		5 linear (512) +1 linear (10)	4 linear with residual and batchnorm (128) + 1 linear (10)	4 linear with residual and batchnorm (128) + 1 GLU (10)	9-conv1d (8~2048, kernel size=9) + 1 linear	Transformer encoder + 4 linear (128)	Transformer encoder + Residual + GLU
Parameters		805,930	53,802	55,092	342,036	833,955	87,462
Loss (MSE)	Train	0.1576	0.1774	0.1656	0.2251	1.2291	1.2503
	Valid	0.1885	0.1767	0.1660	0.2250	1.2632	1.2182
Optimizer		Learning rate exponential decay (learning rate = 0.001 ~ 0.1)					
Epochs		20 (이후 과적합)	150	100	100	100	100
Data	Train	50000	55000	55000	55000	100000	100000
	Valid	10000	30000	30000	30000	30000	30000
Description		과적합, 많은 파라미터	과적합 해결, 파라미터 감소	Residual에 비해 안정적, 빠른 수렴	많은 파라미터, MLP에 비해 높은 loss, 9-bit key까지 가능	6-bit key 공격 실패 (과적합 발생)	더 적은 파라미터 달성, 더 빠르고 안정적인 학습 가능 6-bit key까지만 가능

S-DES 암호 분석

- **Bit type data 및 MLP 구조의 네트워크가 더 좋은 성능을 보임**

  MLP (10-bit key) > CNN (9-bit key) > Transformer encoder (6-bit key)

  → data의 locality 보다는 **global information이 더 중요**함을 알 수 있음

  다시 말해서, 순서, 지역적 정보 보다는 전역적인 정보를 고려할 경우 더 높은 성공률 달성

- CNN은 *pooling, *stride 등의 연산에서 정보 손실 발생 (채널 통해 보완 가능 → **but 파라미터 증가**)

  → **정보 손실을 최소화** 하기 위해 각 레이어의 unit 수를 줄이지 않은 MLP를 사용

- **Residual connection** 적용 시 **parameter 수 현저히 감소** (과적합 감소)

- **Gated Linear Unit** 적용 시 **더 빠른 수렴 속도 및 안정적 학습** (과적합 감소)

- 비트 별 안전성 (Best case)
  - **4번째, 7번째 비트**는 **안전** (60% 미만)
  - **6번째, 9번째, 10번째 비트**는 암호 분석에 **취약** (80%이상)

*pooling : feature map(layer output)의 크기를 줄이거나 conv layer에서 추출된 특징을 강조
*stride : kernel(filter)가 순회하는 간격

S-AES 암호 분석

- 가장 효과적인 MLP with Residual and GLU 신경망을 사용

MLP with GLU		1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th	13th	14th	15th	16th
9-bit	BAP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.7	0.7	0.69	0.69	0.7	0.69	0.69	0.7	0.69
	Dev	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.2	0.19	0.19	0.2	0.19	0.19	0.2	0.19
10-bit	BAP	1.0	1.0	1.0	1.0	1.0	1.0	0.63	0.63	0.63	0.64	0.63	0.63	0.6	0.6	0.6	0.6
	Dev	0.0	0.0	0.0	0.0	0.0	0.0	0.13	0.13	0.13	0.14	0.13	0.13	0.1	0.1	0.1	0.1
11-bit	BAP	1.0	1.0	1.0	1.0	1.0	0.52	0.53	0.52	0.53	0.52	0.52	0.53	0.52	0.51	0.52	0.52
	Dev	0.0	0.0	0.0	0.0	0.0	0.02	0.03	0.02	0.03	0.02	0.02	0.03	0.02	0.01	0.02	0.02
12-bit	BAP	1.0	1.0	1.0	1.0	0.51	0.51	0.5	0.5	0.5	0.5	0.5	0.51	0.5	0.5	0.5	0.5
	Dev	0.0	0.0	0.0	0.0	0.01	0.01	0.0	0.0	0.0	0.0	0.0	0.01	0.0	0.0	0.0	0.0

- **12-bit key까지 모든 비트에 대한 공격이 가능**
  - 모든 비트에 대해 **비슷한 수준의 BAP 달성** (0.8이 넘는 경우도 없음)
  - 1-bit 키 공간이 늘어날 수록 비트별 정확도(BAP)가 약 10%씩 감소하는 추세

*Google Colaboratory (cloud service), Nvidia GPU (Tesla T4, 50GB RAM)
*학습 소요 시간 : 약 8~10시간

S-AES 암호 분석

Network		MLP (12-bit key)
Architecture		Residual + GLU
Layers (units)		10 linear with residual and batchnorm (1024) + 1 GLU (16)
Parameters		11,636,832
Loss (MSE)	Train	0.1826
	Valid	0.1923
Optimizer		Learning rate exponential decay (learning rate = 0.001 ~ 0.1)
Epochs		150
Data	Train	900,000
	Valid	500,000
Description		네트워크 크기를 일정 수준 이상 늘리면 loss는 조금 더 잘 감소하지만, 과적합이 약간 발생 → 데이터 복잡도에 적합한 네트워크 사용하여 과적합을 없애는 방향으로 실험 12-bit key 이상은 현재 상태에서는 암호 분석 불가 (향후 네트워크 튜닝을 통해 정확도 향상 필요)

S-AES 암호 분석

- **모든 비트에 대한 BAP가 비슷**
  → S-DES와 다르게, **특정 취약 비트가 존재하지 않음**
- **S-DES에 비해 더 많은 데이터 샘플과 더 큰 네트워크를 사용**
  → 그러나, **BAP가 0.8 이상인 비트가 존재하지 않음**

- 실험 환경이나 네트워크 및 데이터 크기로 인해서 학습에 너무 오랜 시간이 소요
  → 현재 12-bit key까지 진행되었고, 16-bit key까지 실험해볼 예정
- 12-bit key의 경우 Dev가 음수는 아니지만, 매우 낮은 정확도
  → 하이퍼파라미터 튜닝을 통해 해결할 수 있거나, 12-bit key 이상은 암호 분석이 불가능 할 듯

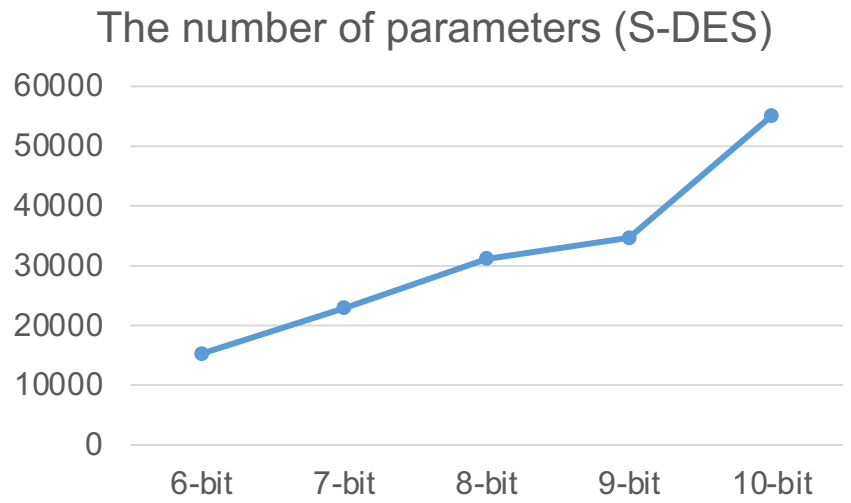
(환경문제로 학습이 끊겨서 다시 진행 중입니다.)

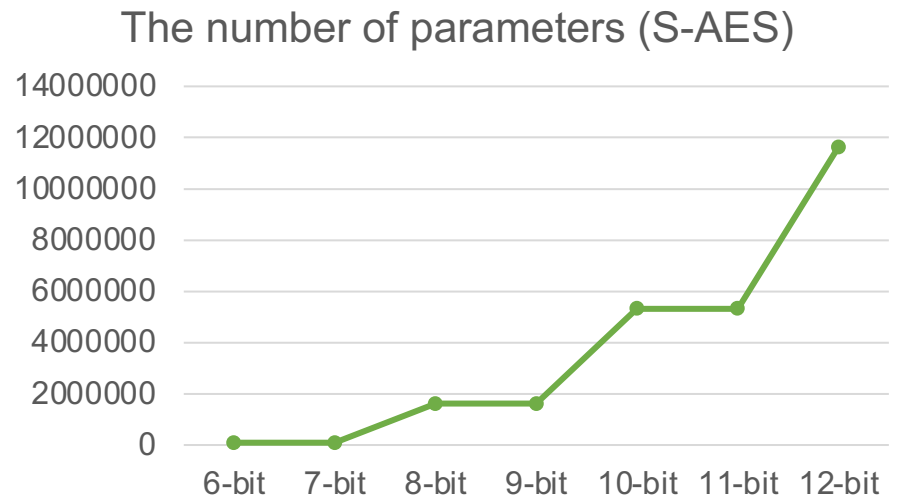
S-DES와 S-AES 비교

- **S-DES**
  - **Initial permutation, Expansion/Permutation, Key addition, S-box(substitution), Swap**으로 구성
- **S-AES**
  - AES와 유사하게 **Substitution nibbles (substitution), Shift rows (permutation), Mix columns(mixing), Key addition**로 구성되며 이를 통해 안전성 제공
  - S-DES에 비해 **키 공간이 큼**
- **S-AES와 S-DES의 비교**
  1. S-AES의 S-box(substitution) 과정이 **S-DES에 비해 혼돈이 큰 것으로 생각**
  → 이를 통해 키와 암호문의 관계가 잘 드러나지 않음
  2. Mix columns 및 permutation을 통해 **확산이 더 잘 이루어짐으로 생각**
  → 평문의 일부분이 암호문의 여러 비트에 영향 → 평문과 암호문 간의 관계가 복잡해짐
  3. **더 큰 키 공간**
  → 안전성 확보
- 즉, S-DES는 S-AES에 비해 **더 적은 키 공간을 사용하고, 확산 및 혼돈이 더 적게 설계되어 특정 취약 비트가 존재하며, 일부 키 비트가 유추 가능**해짐으로 예상

S-DES와 S-AES 비교 - parameters

- 데이터가 복잡해질 수록 네트워크는 커져야 함 (Network capacity 증가)
- **S-DES**
  - 키가 1-bit 증가 시 네트워크의 크기가 큰 폭으로 늘어나진 않음
  - **더 큰 네트워크**를 사용해본 결과, **데이터의 복잡도에 비해 파라미터가 너무 많아서 오히려 과적합 발생**
    → 데이터에 맞는 적당한 복잡도의 네트워크 필요
- **S-AES**
  - S-DES에 사용한 네트워크를 동일하게 사용했을 경우, loss(손실)가 매우 조금씩 감소
    → **네트워크 크기를 늘린 후**에는 조금 더 큰 폭으로 감소, **조금 더 낮은 loss 달성**
    → **S-DES보다 복잡한 네트워크**가 필요 (더 많은 파라미터 필요)





DES, AES로의 확장

- **DES (56-bit key) / AES (128-bit key)로의 확장**
  - 이전 연구들 중 round-reduced simon, speck에 대한 알려진 평문 공격의 경우, text key를 사용
    - **64비트 키이지만 실제 키 공간은 2^{48}**
    - 각 키 비트의 **발생 확률이 다름**
      (1번째 비트는 1.0의 확률로 0이 발생, 2번째 비트는 0.8의 확률로 1이 발생하므로 예측하기 더 쉬움)
  - 56-bit 랜덤 비트키에 대한 암호 분석은 text key에 대한 암호 분석 보다 어려운 작업
    - **56-bit 랜덤 비트키를 사용하는 DES에 대한 공격은 어려울 수**도 있을 것
  - 위와 같이 **Text key를 사용하는 경우 + round reduced에 대해서는 가능할 것으로 생각**
  - 128-bit 랜덤 비트키를 사용하는 AES 또한 어려울 것으로 예상

감사합니다.