QNN 암호 분석

(차분 분석을 위한 quantum neural distinguisher 및 알려진 평문 공격)

https://youtu.be/MvODEAktxCs

Quantum neural distinguisher (암호공모전)

Quantum neural network based Known plaintext attack (암호연구회)

향후 계획

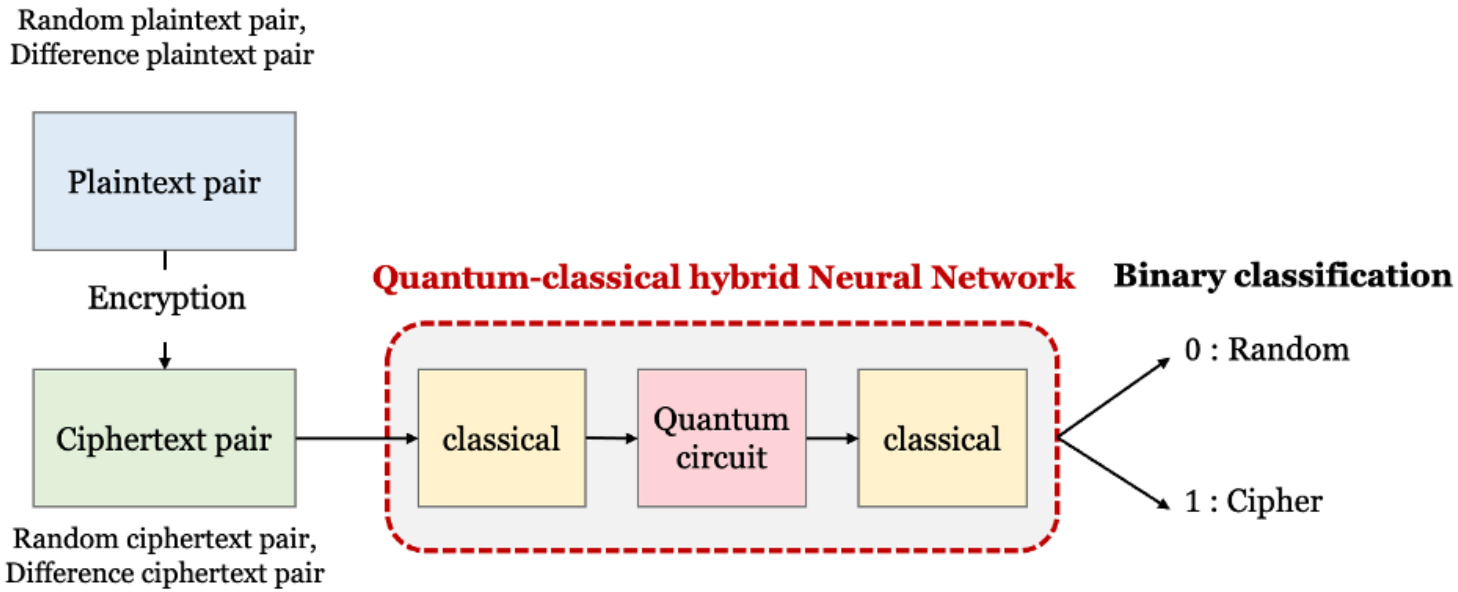
배경 지식

- **차분 분석이나 알려진 평문 공격 및 양자 신경망에 관한 내용**
  - 지난 세미나, 국방부 교육 등에서 언급했기 때문에 생략하겠습니다.

Quantum Neural Distinguisher

Quantum neural distinguisher

- 랜덤 암호문 쌍과 차분 암호문 쌍을 입력
- **랜덤 vs 암호문 분류 (이진 분류)**
- 신경망이 예측한 결과 (정확도)가
  - **0.5보다 큰 값**이 나오면:
    랜덤 데이터가 아닌 입력 차분에 대한 **출력 차분을 갖는 암호문**으로 판단
  - **0.5이하의 값**이 나오면:
    차분이 존재하지 않아 학습 불가능한 **랜덤 데이터 데이터**로 판단
- Neural distinguisher → **차분 공격에 대한 데이터 복잡도 감소!**



Input difference characteristic

- **Input difference**

Input difference

$$P_1 = P_0 \oplus \delta,$$
$$C_0 = E(P_0), C_1 = E(P_1),$$
$$\Delta = C_0 \oplus C_1$$

Output difference

- **S-DES**
  - δ = 0x04
- **S-AES**
  - δ = 0x8000

Dataset preparation

- 차분을 갖지 않는 랜덤 P_0, P_1 선택
- **$P'_0 = P_0 \oplus \delta$: input difference XOR**
- 세 종류의 평문 암호화 → C_0, C_1, C'_0 생성
- **Labeling**
  - $C_0 \,||\, C_1$: 랜덤 암호문 쌍 → label 0
  - $C_0 \,||\, C'_0$: 차분 암호문 쌍 → label 1
- Dataset에 추가

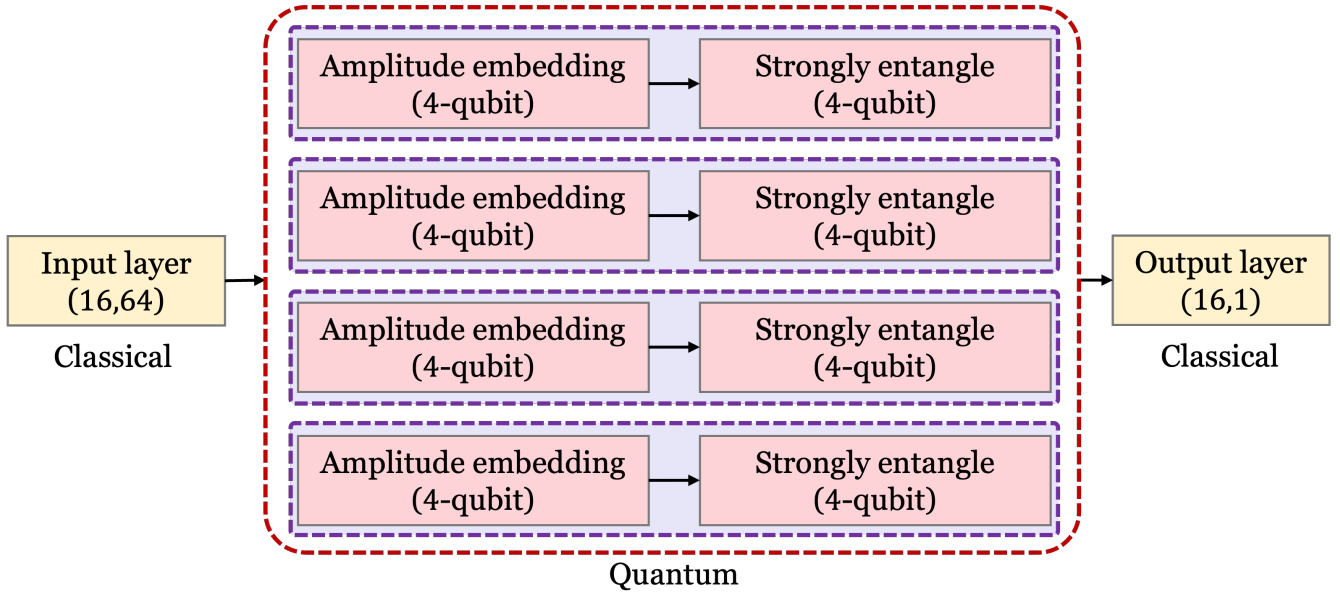
Algorithm 1 Dataset preparation

Input: Input difference (δ), The number of data (N_{ds}), Encryption function(Enc)
Output: Dataset (DS)

for $i = 1$ **to** $N_{ds} \div 2$ **do**
  Choose random 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 *class* 0 (Random ciphertext pair)
  $(C_0||C'_0)$ is labeled *class* 1 (Difference ciphertext pair)
  $DS_i \leftarrow (C_0||C_1)$
  $DS_{i+(N_{ds} \div 2)} \leftarrow (C_0||C'_0)$
end for
return DS

Design of Quantum Neural Distinguisher

- **하이브리드 신경망 구조**
  - **Qubit가 비교적 적게 필요**한 **Amplitude embedding circuit** 사용
    - 4-qubit 회로를 사용 → 16개의 값을 하나의 회로에 임베딩 가능
  - 다른 회로보다 안정적인 성능을 갖는 **Strongly entangle circuit** 사용
  - 입출력 레이어는 classical layer 사용
    - 입력은 8-bit 평문 (S-DES) 쌍일 경우 16-bit → 16 neuron
    - 출력은 이진 분류 → 1 neuron



Training

- 사용할 회로의 수 (N_{qc}) = 이전 레이어의 뉴런 수 $(Neuron_H)$ / 한번에 임베딩 할 수 있는 데이터 길이 $(2^{N_{qubit}})$
- $Q_{amp_{(i)}}$ = i 번째 임베딩 회로 $(i < N_{qc})$, $Q_{states_{(i)}}$ = i 번째 회로의 qubit들의 상태

- **Epoch만큼 아래 과정을 반복 수행**
  - **Input → hidden → quantum circuit → output → loss → parameter update**
  - **Input → hidden → quantum circuit**
    1. **이전 레이어의 뉴런을 각 회로에 나누어 임베딩**
       → 이전 레이어의 출력 64개를 16개씩 나누어 4개의 4-qubit circuit에 할당
    2. 임베딩 된 후,i번째 회로의 큐비트 상태를 **i번째 $Q_{ent_{(i)}}$에 입력**
    3. $Q_{ent_{(i)}}$의 출력을 **measure**
  - **quantum circuit → output**
    1. 4개의 양자 회로의 **출력들을 연접한 후, 출력 레이어에 입력**
    2. 신경망의 최종 출력 얻음
  - **loss → parameter update**
    1. Binary crossentropy 손실 함수 사용 (이진 분류)
    2. 이를 통해 loss, accuracy 계산 및 **회로의 파라미터 갱신**
  - **정확도가 0.5보다 크다면, 차분을 갖는 데이터를 분류할 수 있는 모델**
    - **Quantum neural distinguisher로 사용**

Algorithm 2 Training process using quantum-classical hybrid network

Input: Dataset (DS), The number of qubits (N_{qubit}), Classical input, hidden and output layer $(Input, H, Output)$, Quantum circuit for amplitude embedding (Q_{amp}), Quantum circuit for quantum layer (Q_{ent})
Output: Trained hybrid model (QC_{Hybrid})

$Neuron_H \leftarrow$ the number of neuron of hidden layer
$H_i \leftarrow i$-th neuron of classical hidden layer
$k \leftarrow 2^{N_{qubit}}$
$N_{qc} \leftarrow (Neuron_H \div k)$; the number of quantum circuit
$Q_{amp_{(i)}}$ is i-th Q_{amp}
$Q_{ent_{(i)}}$ is i-th Q_{ent}
$Q_{states_{(i)}} \leftarrow$ states of qubits of i-th quantum circuit

for $i = 0$ **to** $Epoch - 1$ **do**
  $x \leftarrow Input(DS)$
  $x \leftarrow H(x)$
  for $i = 0$ **to** $N_{qc} - 1$ **do**
    $Q_{states_{(i)}} \leftarrow Q_{amp_{(i)}}(H_{k*i+0}, H_{k*i+1}, \cdots, H_{k*i+15})$
    $Q_{states_{(i)}} \leftarrow Q_{ent_{(i)}}(Q_{states_{(i)}})$
    $x_i \leftarrow \text{measure}(Q_{states_{(i)}})$
  end for
  $x \leftarrow (x_0 || x_1 || \cdots || x_{N_{qc}-1})$
  $outputs \leftarrow Output(x)$
  Compute $loss$ and $accuracy$
  Adjust the parameters of the quantum circuit
end for
if $accuracy < 0.5$ **then**
  Abort QC_{Hybrid}
else if $accuracy > 0.5$ **then**
  return QC_{Hybrid}
end if

Quantum layer - amplitude embedding

- 임베딩 회로
  - 변경될 수 있는 파라미터가 아니라, **입력된 고전 데이터가 파라미터로 쓰임**
- RY 회전 게이트와 얽힘을 위한 $CNOT$ 게이트로 구성
- 16개의 데이터를 큐비트의 **상태 벡터 (길이가 16)로 임베딩**
  - 64개의 입력 데이터가 있다면, amplitude embedding circuit 4개 필요

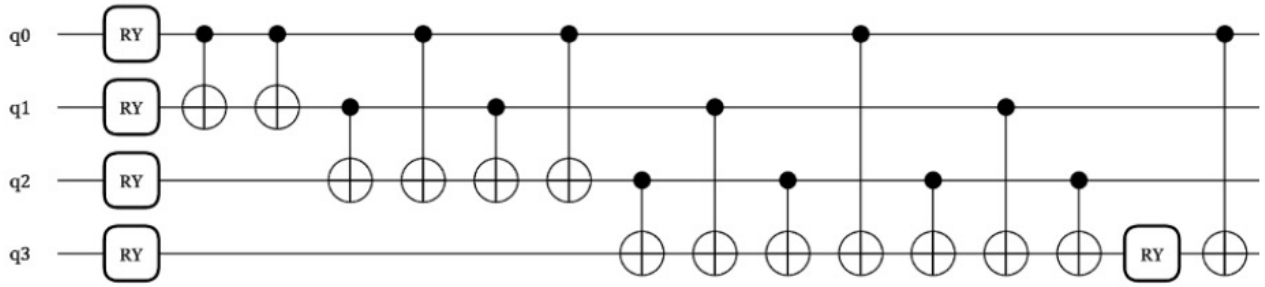


Fig. 3: Quantum circuit for amplitude embedding.

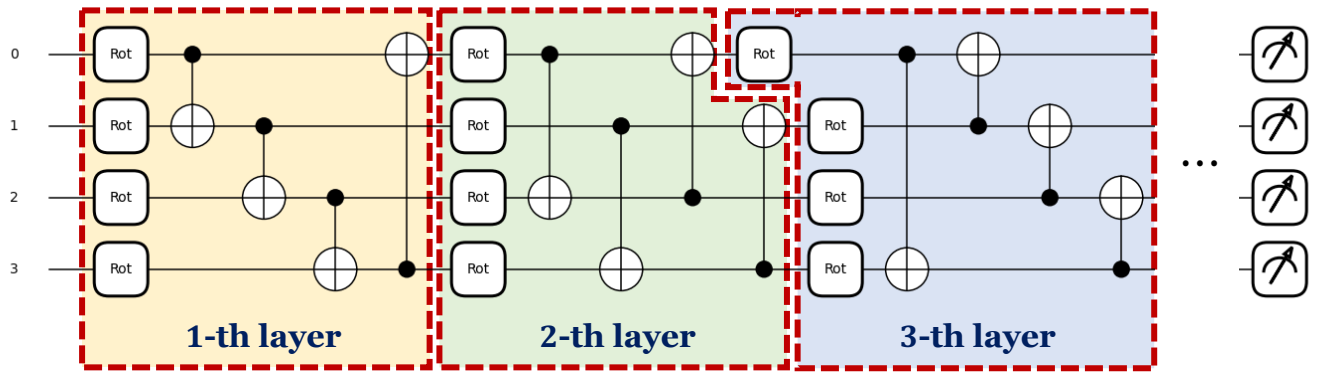
Quantum layer – strongly entangling circuit

- **Rot 회전 게이트** 사용
  - 각 레이어에는 N_{qubit} 만큼의 Rot 게이트 사용
  - $Rot = Rz\ Ry\ Rz$ 조합 → Rot 게이트 수의 3배만큼의 파라미터가 필요
- **레이어 반복 횟수 (r)는 r번째 레이어의 얽힘에 영향**
- N_{qubit}개의 qubit을 사용하는 회로의 l번째 레이어에 대해

$$r = l \bmod N_{qubit}$$

- **r번째 레이어에서, i번째 qubit과 $(i + r \bmod N_{qubit})$번째 qubit이 얽힘**

→ 레이어가 쌓일 수록 더 다양한 얽힘 생김



실험 결과 1

- 차분분석을 위한 quantum neural distinguisher 실험 결과
  - **두 암호에 대해 Qubit 수, embedding layer와 quantum layer 종류, epoch 동일**
    - 2-qubit은 성능 미달, 8-qubit은 너무 느린 실행 속도
  - 전체 파라미터의 수는 고전 신경망의 파라미터 수까지 더해진 값
  - **회로의 파라미터 수는 the number of rotation gate 와 동일**

$$N_{Rotation} = N_{qc} \cdot (3 \cdot (N_{qubit} \cdot N_{ql}))$$

회전 게이트의 수 구하는 수식

- **S-DES**
  - 양자 회로 1개
  - 양자 레이어 수 15개
  - 회전 게이트의 수 = 1 x 3 x 4 x 15 = 180
  - 학습 데이터 수 1000개
  - **Test accuracy : 98%**
- **S-AES**
  - 양자 회로 4개
  - 양자 레이어 수 10개
  - 회전 게이트의 수 = 4 x 3 x 4 x 10 = 480
  - 학습 데이터 수 2000개 (S-DES에 비해 많은 데이터 필요)
  - **Test accuracy : 99%**

Table 1: Details of quantum-classical hybrid neural network for differential cryptanalysis.

	S-DES	S-AES
The number of qubits	4	4
Quantum embedding	Amplitude	Amplitude
Quantum layer	Strong entangle	Strong entangle
The number of quantum circuit	1	4
The number of quantum layer	15	10
The number of rotation gate	180	480
The number of parameters	457	777
The number of data	1000	2000
Epoch	25	25
Test accuracy	98%	99%

실험 결과 1

- **S-AES가 S-DES에 비해 더 많은 데이터와 회로 수 그리고 더 많은 파라미터 필요**

- 하이브리드 신경망 사용
  - **진폭 임베딩**을 사용하고 quantum-only 구조를 사용하지 않음으로써 **필요한 큐비트 수 줄임**
  - **히든 레이어를 더 줄이면, 큐비트를 더 절약**할 수 있음

- Strongly entangling layer가 random, basic layer에 비해 **회전 게이트 및 얽힘이 많고 정해진 구조**

  → **더 안정적임**

- 입력 차분을 다른 값으로 설정할 경우, **0.5에 가까운 값이 나옴**

실험 결과 2

- 고전 신경망과 비교한 결과, **같은 epoch과 데이터 수에 대해 더 높은 정확도, 더 적은 파라미터 달성**
  - **S-DES**
    - 데이터 수 1000개에 대해 quantum이 **2% 더 높은 정확도 달성**
    - 고전 distinguisher에 비해 **파라미터 수 28.7% 감소**
  - **S-AES**
    - 데이터 수를 1000개로 하면 quantum이 **18% 더 높은 정확도** 달성, **파라미터 수 43% 감소**
    - 데이터 수를 2000개로 하면 quantum이 **1% 더 높은 정확도** 달성, **파라미터 수 28.6% 감소**
  - **이러한 quantum advantage를 얻을 수 있는 이유**
    - Qubit의 **데이터 표현 범위가 더 풍부** (블로흐 구면 위의 모든 점을 표현 가능)
      → 정확도 향상
    - 모든 노드가 연결된 classical MLP와 다르게 quantum circuit은 **특정 수만큼의 게이트와 파라미터를 요구**
      → 파라미터 감소

Table 2: Comparison between classical and quantum classical neural networks for differential crypanalysis for S-DES.

$(Epoch, N_{Data})$	(25, 1000)	
Target	S-DES (Classical)	S-DES (Quantum, Ours)
Tr	96	97
Val	97	97
Ts	96	98
N_{Params}	641	457

Table 3: Comparison between classical and quantum classical neural networks for differential crypanalysis for S-AES.

$(Epoch, N_{Data})$	(25, 1000)		(25, 2000)	
Target	S-AES (Classical)	S-AES (Quantum, Ours)	S-AES (Classical)	S-AES (Quantum, Ours)
Tr	68	92	92	100
Val	75	86	99	99
Ts	65	83	98	99
N_{Params}	1089	617	1089	777

향후 계획

- S-present, PIPO, CHAM에 대한 quantum neural distinguisher 구현
  → 부채널 공모전

Quantum Neural Network based known plaintext attack

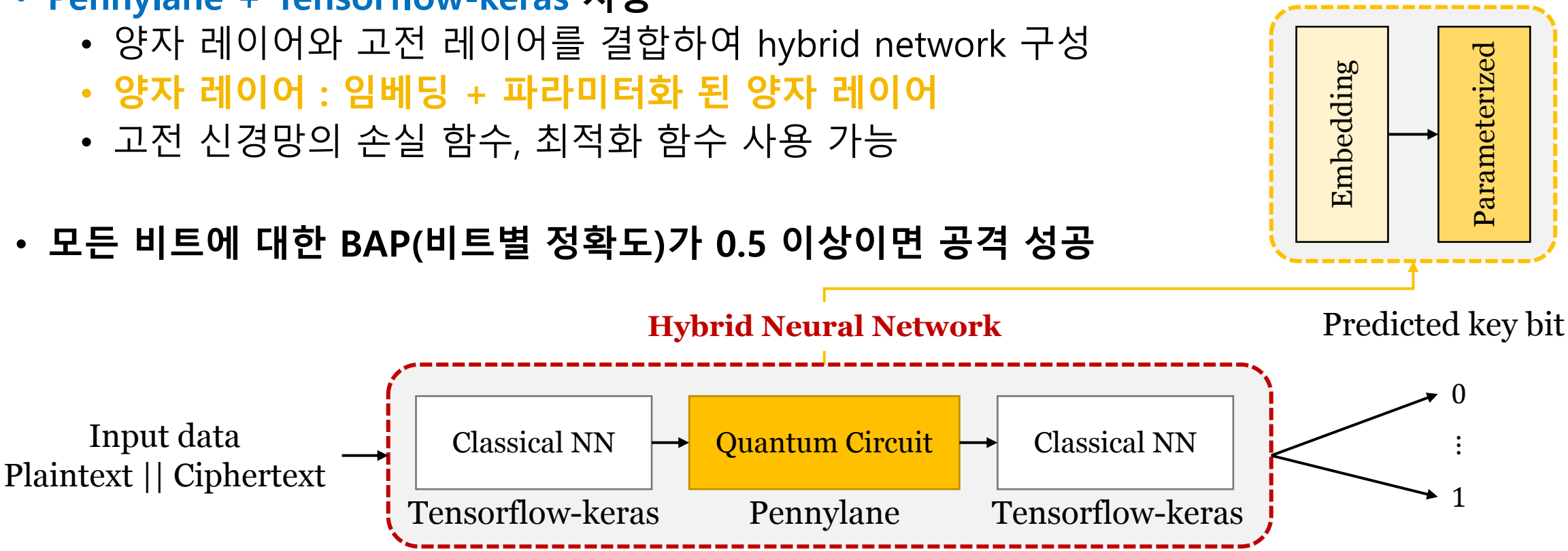
Quantum-classical hybrid NN을 이용한 알려진 평문 공격

- **Dataset**
  - (plaintext, ciphertext) 쌍 그리고 key를 **비트로 표현**
  - **Input data** : (plaintext, ciphertext) bit
  - **Label** : key bit

Input data						Label		
Plaintext bit			Ciphertext bit			Key bit		
0	…	1	1	…	1	1	…	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

- **Pennylane + Tensorflow-keras 사용**
  - 양자 레이어와 고전 레이어를 결합하여 hybrid network 구성
  - **양자 레이어 : 임베딩 + 파라미터화 된 양자 레이어**
  - 고전 신경망의 손실 함수, 최적화 함수 사용 가능

- **모든 비트에 대한 BAP(비트별 정확도)가 0.5 이상이면 공격 성공**



Quantum-classical hybrid NN을 이용한 알려진 평문 공격

- **Hybrid network**
  - **Qubit 절약, 학습 시간 절감, quantum-only 보다 더 안정적인 성능**
- **Library**
  - **Tensorflow-keras**와 **pennylane** 결합
- **Device**
  - Pennylane의 default.qubit (**default simulator**), default.mixed (**noise simulator**, 실험 예정)
- **Quantum circuit**
  - **Amplitude layer**
    - **qubit 절약**하기 위해 사용
  - **Random, Strongly entangling**
    - **Random**은 얽힘이 다양
    - **Strongly**는 회전 게이트가 3배, 레이어를 쌓을 수록 얽힘이 강함
    - **Basic** circuit은 두 가지 특징을 모두 가져서 제외 (얽힘이 너무 단순 + 회전 게이트 적음)
- **Shots = 1000**
- **이 외의 변경 가능한 요소**
  - Quantum layer 수, circuit 수, qubit 수, data 수, 고전 레이어 수 변경하며 실험

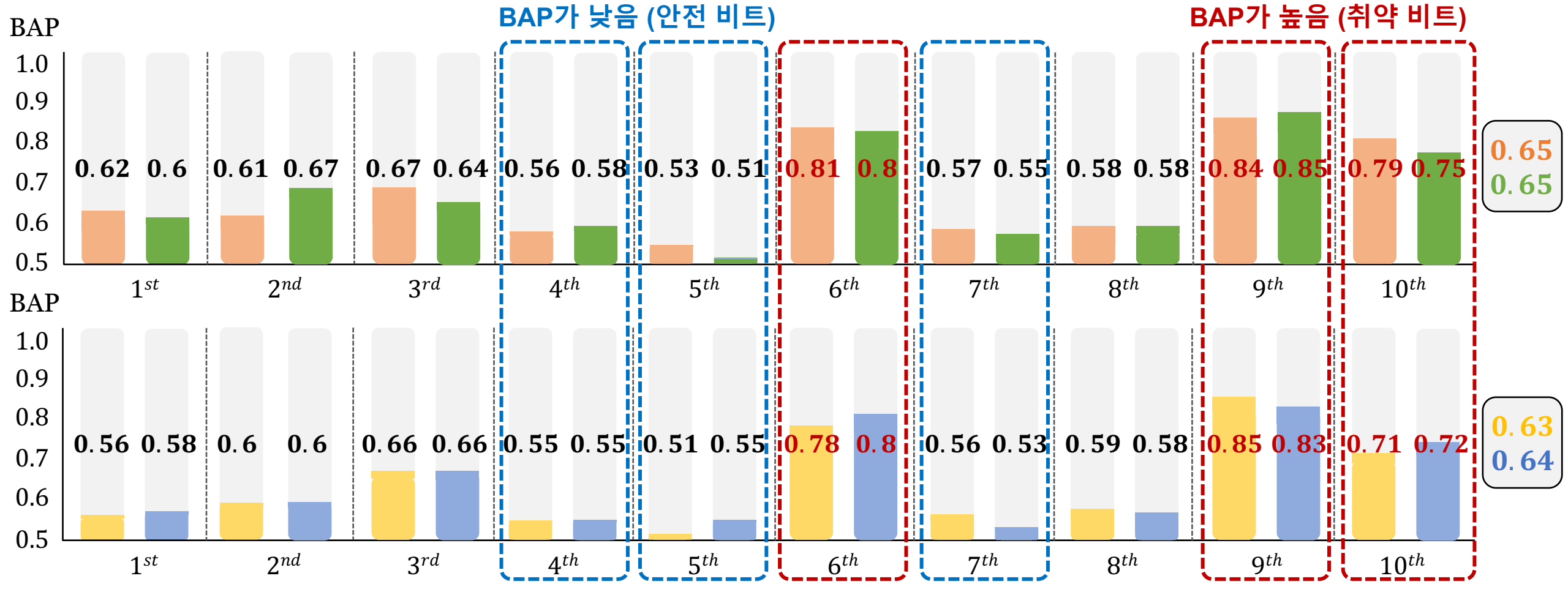
Details of quantum-classical hybrid neural network

	Quantum (Random)	Quantum (Strongly)	Description
Quantum circuit	Amplitude + Random	Amplitude + Strongly	**Qubit 절약** 위해 amplitude embedding **다양한 얽힘** (random) **더 많은 회전 게이트와 강한 얽힘** (strongly)
# of qubit	4	4	2-qubit은 충분한 성능 X 8-qubit은 실행 시간이 많이 소요 따라서, **4-qubit이 적절**
# of quantum layer	10	10	5-bit key까지는 5개의 quantum layer 가능 그 이상은 **10~15개** 적용 필요 → **1 epoch에 약 2~3만 초** 20개는 학습 소요 시간 매우 증가 → **1 epoch에 약 4~5만 초**
Architecture of classical hidden layer	128, 128, 128, 32	128, 128, 128, 64	실험 통해 적절하게 설정
# of circuit	2	4	Classical hidden layer의 구조와 # of qubit에 의해 결정 **2~4개가 적당**한 것으로 생각 더 늘릴 경우 학습 소요 시간 매우 증가
# of parameters	43956	44276	**Quantum circuit의 parameter는 매우 적으므로** classical layer가 많을 수록 크게 증가 Classical NN의 # of parameter : 55092
# of data	28500	19950	-

Classical vs Quantum

Classical (MLP, # of data : 28500, # of param : 55092)
Quantum (Random, # of data : 28500, # of param : 43956)
Classical (MLP, # of data : 19950, # of param : 55092)
Quantum (Strongly, # of data : 19950, # of param : 44276)

- Quantum NN의 학습 시간이 매우 오래 걸려서, classical과 quantum에 대해 **25 epoch만 학습**한 후 결과 비교
- **Random**과 **Strongly entangling** layer에 대해 실험 → **회로 개수, quantum layer 수 변경하며 실험 중**
- Classical과 quantum NN이 **거의 비슷한 성능을 보이지만, 파라미터 수는 10000개 이상 적음**
- **데이터 수가 상대적으로 적은 경우 (19950)**, quantum이 평균적으로 **1% 더 높은 정확도 달성**



결론

- **양자 인공지능을 위한 다양한 양자 컴퓨팅 환경 및 SDK 분석**
  - 시뮬레이터, 하드웨어, 회로 동작 시간, 지원 라이브러리 등을 비교 분석
  - 이에 따라 양자 인공지능 기반의 **암호 분석에 적합한 플랫폼 선택**
- **Pennylane과 tensorflow-keras를 활용하여 quantum-classical hybrid network 기반의 암호분석 수행**
- 현재는 **noise가 없는 simulator** 사용
- **Quantum layer에 대한 여러 요소들을 고려하여 실험** 진행
- **Quantum advantage 얻음**
  - 데이터가 상대적으로 적은 경우, quantum 방식이 **더 적은 파라미터로 1% 더 높은 정확도**를 달성
  - 조금 더 많은 데이터를 사용한 경우, quantum 방식이 **더 적은 파라미터로 동일 정확도** 달성
- 그러나, **학습에 소요되는 시간이 매우 크다는 한계점 존재**

어려움..

- **가속화 시뮬레이터 (Lightning.qubit)**
  - **Strongly entangling layer**에는 역전파에서의 자동 미분을 지원하지 않는다는 **에러 발생하여 사용 불가**

- 학습 시간이 너무 오래 걸려서 다양한 실험을 해보기가 어려움
  - 데이터를 1차 결과물의 약 반정도 (3만개 미만)를 사용하여도 **1 epoch에 2~3만 초 정도 소요**

- 30 epoch까지도 잘 돌아가다가 **갑자기 layer 에러가 발생**하는 경우 존재 (로컬)
  - 그러나, 다시 실행하거나 다른 컴퓨터에서 실행시킬 경우 **같은 코드여도 에러가 발생하지 않기도 함**
    → 아직 원인 파악은 하지 못했습니다..

***교수님 혹시 여기를 보고 계시다면, 이 내용을 암호연구회 발표 자료에 추가해야 할지 여쭤봐도 될까요..**

향후 계획

- **Noise simulator** (default.mixed) 사용
  → **양자 컴퓨터의 noise를 고려**
  → 그러나 Random layer에만 적용 가능

- **다른 암호 분석 (S-AES 등) 후, classical NN과의 비교**
  - 현재, classical 보다 **더 적은 파라미터 사용**
  - **동일 정확도** (# of data : 28500) 및 **1% 더 높은 정확도** (# of data 19950) 달성
  - **# of parameter 및 정확도에 대한 quantum advantage 확인 완료**

  - 모델 최적화, 데이터 수 증가, epoch 증가 실험
    → 1차 결과물과 동일한 조건에 대한 **quantum advantage를 얻을 수 있는지 확인**

- **Quantum AI를 활용한 암호 분석의 한계점 도출**

감사합니다.