YONSEI UNIVERSITY

양자컴퓨팅입문

VQA ∋ Quantum Machine Learning

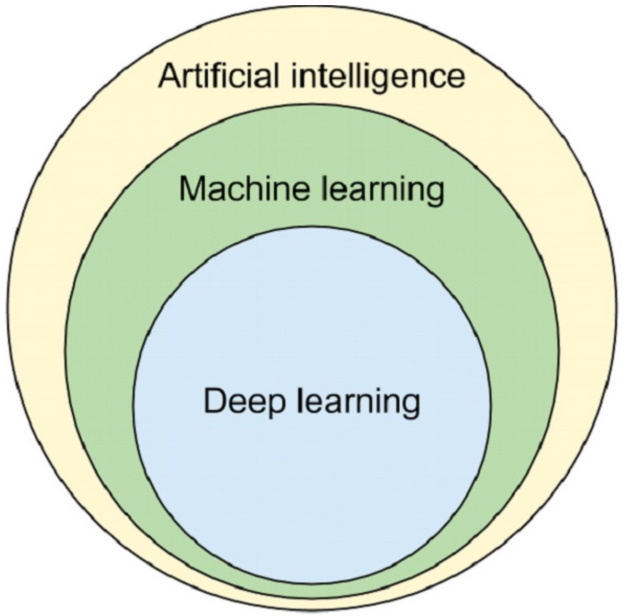
최석원

목차

Machine learning

빅데이터를 스스로 분석,결론

- 복잡한 패턴에 대한 학습을 통해 데이터 예측
- Unsupervised Learning
- Supervised Learning
- Reinforcement Learning
- Deep Learning 등등



Quantum Machine Learning (QML)

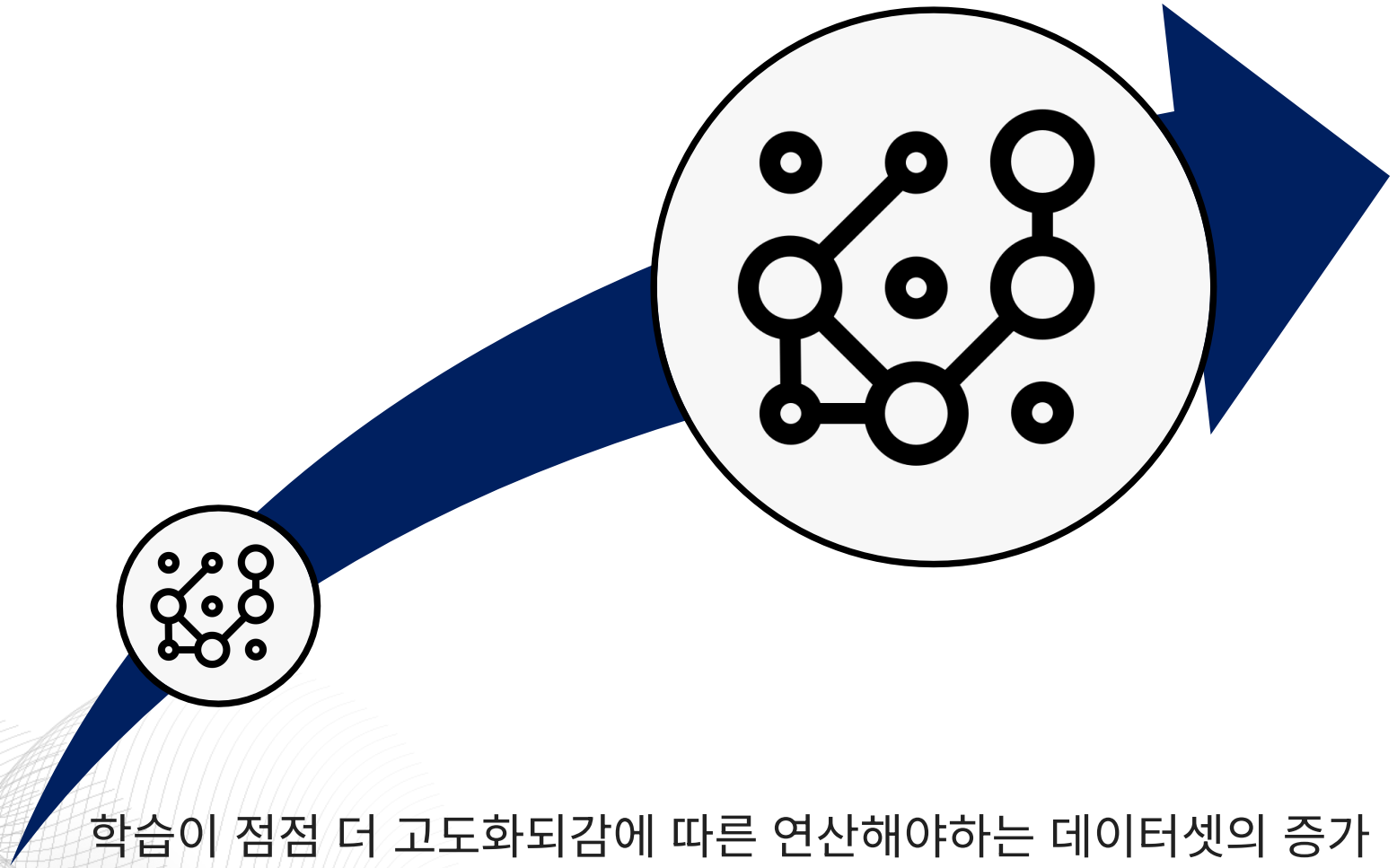
Machine Learning and Quantum Computers

둘의 장점을 활용한 ML 알고리즘

- VQE, QAOA, QSVM, QNN, QPCA, Quantum k-Means Clustering 등 여러 알고리즘 존재

- 이 강의에서는 QCNNs을 다룰 예정

Why Quantum Machine Learning?

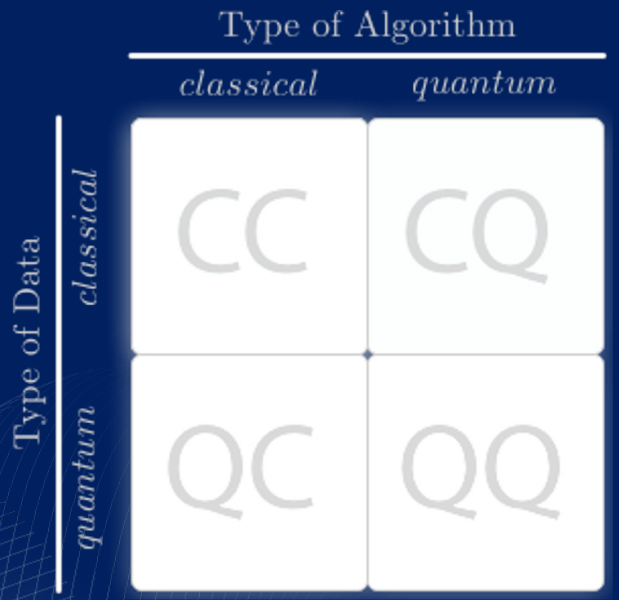


학습이 점점 더 고도화되감에 따른 연산해야하는 데이터셋의 증가

Quantum Machine Learning (QML)

데이터의 종류와 알고리즘의 종류

- 데이터의 방법과 처리방식의 분류
- 여러 방법이 존재한다.
- 경우에 따라 클래식 데이터를 양자 데이터로 변환



Data encoding

클래식 데이터를 양자 상태에 인코딩하는 다양한 방법들
양자상태로 변환하여 양자연산하기 위해서는 필수적

Basis Encoding

- 데이터를 큐비트의 상태$|0\rangle$ or $|1\rangle$ 으로 매핑

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 11 \\ 01 \\ 00 \\ 11 \end{bmatrix} = \begin{bmatrix} |11\rangle \\ |01\rangle \\ |00\rangle \\ |11\rangle \end{bmatrix}$$

Data encoding

Amplitude encoding

- 양자 상태의 진폭으로 인코딩하는 방식
- 고차원의 데이터를 효율적으로 인코딩 가능

$$|\psi\rangle = \sum_i x_i |i\rangle,$$

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{19} \\ 1/\sqrt{19} \\ 0/\sqrt{19} \\ 3/\sqrt{19} \end{bmatrix}$$

Data encoding

Angle encoding

- 데이터를 큐비트의 회전 각도로 인코딩하는 방법

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{19} \\ 1/\sqrt{19} \\ 0/\sqrt{19} \\ 3/\sqrt{19} \end{bmatrix}$$

$$|x\rangle = \bigotimes_{i=1}^{N} \cos(x_i)\,|0\rangle + \sin(x_i)\,|1\rangle$$

Data encoding

Many ways of encoding

Encoding	Qubits	State prep runtime
Basis	nN	$O(N)$
Amplitude	$\log(N)$	$\frac{O(N)}{O(\log(N))}$
Angle	n	$O(N)$

Data encoding – Currently Studied

Hamiltonian encoding

해밀토니안(Hamiltonian)을 데이터에 맞춰 변환하여 표현하는 방식

Methods of Hamiltonian encoding

해밀토니안을 양자 컴퓨터가 처리할 수 있도록 큐비트에 맞춰 변환

$$|\psi\rangle = \sum_{i}^{dim} f(x)\,\sigma_i$$

파울리 매트릭스를 활용하여 해밀토니안으로 인코딩을 진행

자세한 식은 Appendix –A 참조

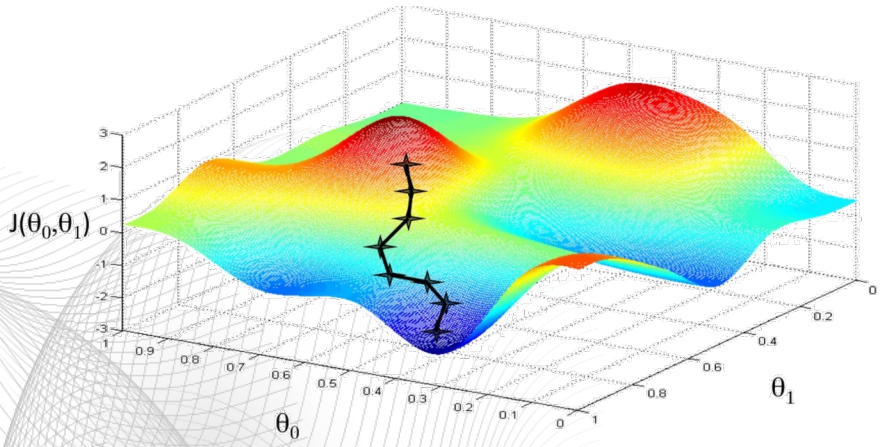
Theorical Background – Machine learning

Gradient Descent

- $loss\ function$의 각 파라미터에 대한 기울기를 계산
- $loss\ function$이 감소하는 방향으로 파라미터를 업데이트하는 최적화 알고리즘.

- Batch Gradient descent
- $\theta_j \leftarrow \theta_j + \alpha \sum_{l=1}^{n} \left(y^{(i)} - h_\theta\left(x^{(i)}\right) \right) x_j^{(i)}$

Aggregate the update of θ for all training examples



Theorical Background – VQA

Variational Quantum Algorithm

- 양자역학의 Variational Method에서 영감을 받아 제안된 알고리즘
- $\langle\psi|H|\psi\rangle \geq \lambda_{min}$ 를 활용하여 수렴값을 찾을 수 있지 않을까?
- $\min_{\theta}\langle x\left|U^{\dagger}(\theta)HU(\theta)\right|x\rangle$ 을 찾는 과정이라 볼 수 있다.

Theorical Background

Parameter-shift rule

$\min_{\theta}\langle x|U^{\dagger}(\theta)HU(\theta)|x\rangle$의 값을 찾기 위해..

- 양자 회로에서는 파라미터에 대한 미분을 고전적으로 계산하기 어렵다.
- 따라서 파라미터의 특정 두 가지 변화값을 사용하여 미분을 계산하는 방식으로 연산
- $let\ U_i(\theta_i) = e^{i\theta_i G}$
- $f(x;\theta) = \left\langle 0 \left| U_0^{\dagger}(x)U_i^{\dagger}(\theta_i)\hat{B}U_i(\theta_i)U_0(x) \right| 0 \right\rangle = \left\langle x \left| U_i^{\dagger}(\theta_i)\hat{B}U_i(\theta_i) \right| x \right\rangle$
- $then\ \nabla_{\theta_i} f(x;\theta) = \frac{[f(x;\theta+s)-f(x;\theta-s)]}{2\sin(\Omega s)}\Omega$

- 자세한 유도는 Appendix – B 참조

YONSEI UNIVERSITY

“

그래서 이 성질이 왜 놀랍죠?

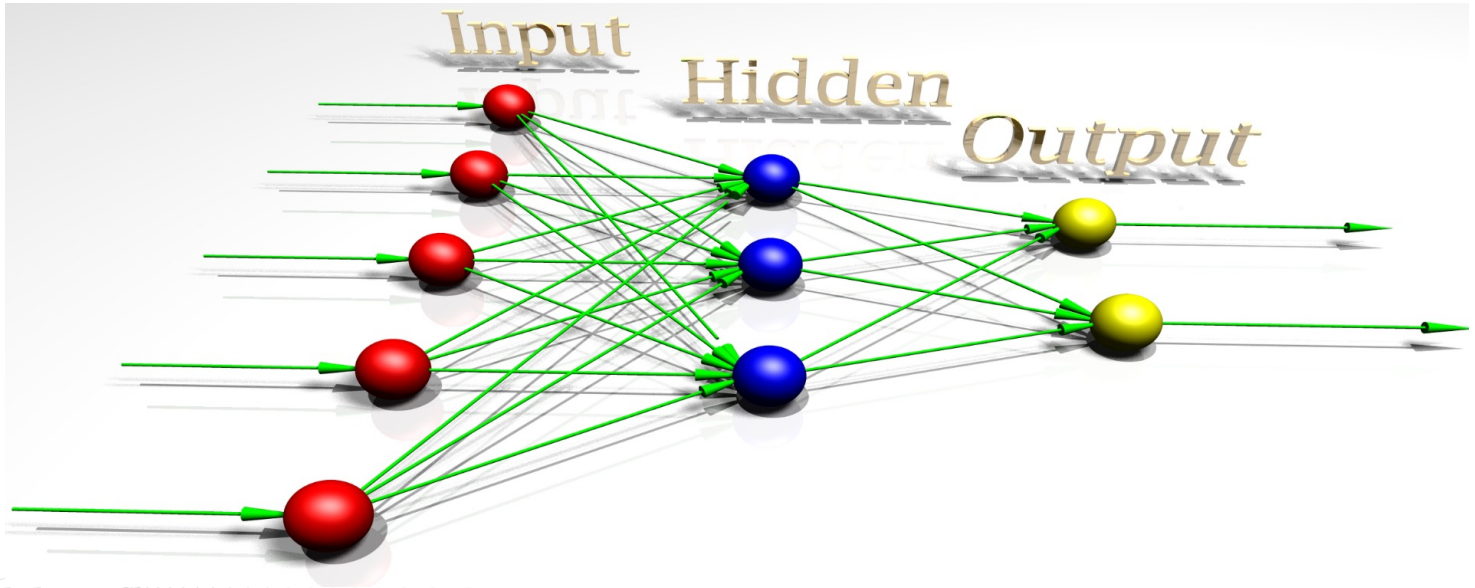
Gradient 를 통해 최적화가 가능한 것입니다.
이것의 응용 분야로 QML, 그중 QCNN이 있죠

”

Theorical Background

Classical feed-forward neural networks

- input, hidden, output layer로 된 여러층의 뉴런들로 구성



- Perceptron 과 non-linear activation function를 가지고 있음
- Backpropagation과 Gradient Descent 방법을 통해 가중치를 업데이트(학습)

Quantum perceptron

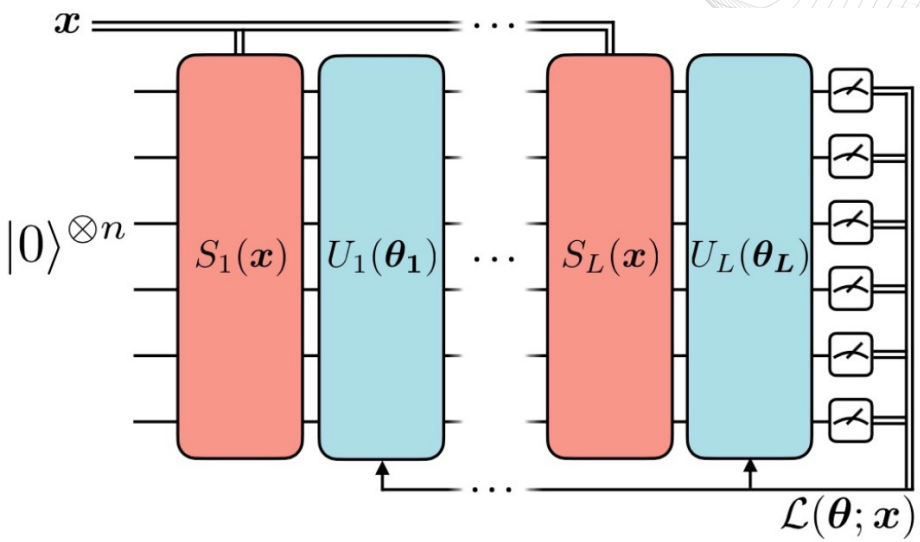
특징

- 전통적인 퍼셉트론의 기능을 양자 원리를 활용하여 확장
- 더 복잡한 데이터의 패턴을 학습

양자 형태로 연산

- 입력 데이터를 큐비트 형태로 인코딩
- 전통적인 퍼셉트론에서 가중치를 곱하고 더하는 것과 유사
- 양자 퍼셉트론은 큐비트를 통해 여러 입력을 동시에 처리

Quantum neural networks



의미

많은 수의 parameter 계산을
Quantum perceptron 통해 빠르게 계산

기존방식과의 차이

- multiple data re-uploading 이 가능한 모델도 존재
다른 양자 gate로 반복해서 데이터를 업로드 가능

- universal quantum classifier
단일 큐비트에서도 고전적인 방법의 도움을 받으면
QNN이 함수 근사치 연산 가능

Multiple data re-uploading

주어진 데이터를 여러 번 회로에 업로드하여 더 복잡한 특성을 학습 수 있게 하는 기법
더욱 복잡한 함수나 패턴을 근사 가능

QNN에서 data re-uploading

데이터를 여러 번 반복하여 회로에 입력하는 과정이므로,
이를 통해 Fourier 계수를 반영하는 방식으로 변환 가능

$$x_1 = \sum_{-\infty}^{\infty} c_{k,1} e^{i2\pi kt}$$: 첫번째 업로드

$$x_{final} = \sum_{n=1}^{N} \sum_{k=-\infty}^{\infty} c_{k,n} e^{i2\pi kt}, \qquad |\psi\rangle = \sum_{n=1}^{N} \sum_{k=-\infty}^{\infty} c_{k,n} |k\rangle$$

주파수 성분이 data re-uploading 과정에서 양자 회로에 입력되어,
최종적으로 discrete Fourier series 로도 생각 가능
=> Machine Learning의 Optimization Process로 볼 수 있다.

Convolutional neural networks (CNNs)

구조

Convolutional Layer, Activation Function, Pooling Layer,
Fully Connected Layer등으로 구성

학습 과정

- 입력 이미지가 CNN을 통과하며 Convolutional, Pooling Layer 를 거쳐 특징을 추출
- 대량의 라벨링된 데이터셋을 통해 학습, loss function 최소화
- Backpropagation과 Gradient Descent 통해 학습

Convolutional neural networks (CNNs)

Convolution 연산

입력 데이터에서 지역적인 특징을 추출하는 과정
주로 이미지의 특정 패턴(예: edge나 texture)을 인식하기 위해 필터(또는 커널)를 사용
이 필터는 입력 이미지의 각 위치에 적용되어 작은 부분을 스캔,
각 부분마다 필터와의 합성곱(convolution)을 수행하고 결과를 새로운 피처 맵으로 저장

$$y_{i,j} = \sum_{m,n} x_{i+m,j+n} \cdot k_{m,n}$$

: 입력 데이터 x 에 커널 k 를 적용하여 특징 맵 y 생성

Pooling 연산

최대 풀링(Max Pooling): 특징 맵 내 가장 큰 값만 유지하여 데이터 압축
평균 풀링(Average Pooling): 영역 내 평균값 사용

Quantum convolutional neural networks (QCNNs)

Convolution 연산

QCNN에서는 양자 게이트를 사용하여 데이터를 변환

여기서 필터는 고정된 수학적 구조가 아니라, 특정 양자 상태와 양자 연산을 통해 생성
이렇게 생성된 양자 상태는 이미지의 특정 패턴에 대응

Pooling

양자 CNN의 pooling은 quantum state reduction 과정을 통해 수행
classical CNN에서는 단순히 max 또는 average로 피처 맵의 크기를 축소,

QCNN에서는 양자 중첩 및 얽힘상태를 이용해 특정 qubit 정보만을 남기고
다른 qubit를 측정하거나 제거함으로써 데이터를 압축

CNN과 달리 양자 데이터의 축소는 중첩된 상태에서 중요한 정보만 남기며
최종 피처 맵에 영향

QCNN의 Convolution VS CNN의 Convolution

클래식 Convolution

입력 데이터에 커널을 선형적으로 적용, 행렬곱 기반 계산

양자 Convolution

양자 상태 간의 inner product를 통해 특징을 추출
특징 추출 과정이 양자 회로를 통해 구현

$$Quantum\ Convolution: \left|\psi_{output}\right\rangle = U(\theta)\left|\psi_{input}\right\rangle$$

QCNN은 파라미터화된 양자 게이트와 내적 연산을 통해 비선형적 특징을 표현

Classical Pooling vs. Quantum Pooling

클래식 Pooling

- 최대/평균 풀링을 통해 정보 압축 및 연산량 감소
- 입력의 특정 부분을 대표 값으로 단순 변환

양자 Pooling

양자 상태의 중첩을 이용한 정보 압축 방식
대표적 양자 풀링 방법:

1. 양자 측정을 통해 확률적으로 데이터 차원 축소
2. 특정 큐비트만 관측하여 전체 양자 상태를 축소하는 방식

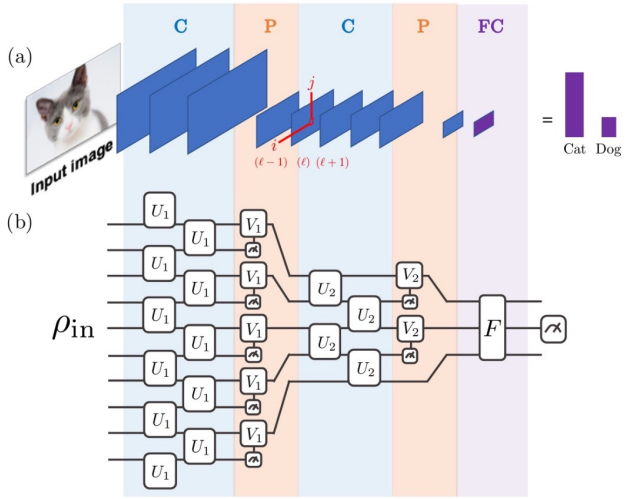
$$Quantum\ Pooling : \rho_{pooled} = Tr_B(\rho)$$

특정 부분집합 B 를 trace-out하여 얻은 축소 밀도 행렬로 풀링을 진행했다

Quantum convolutional neural networks (QCNNs)

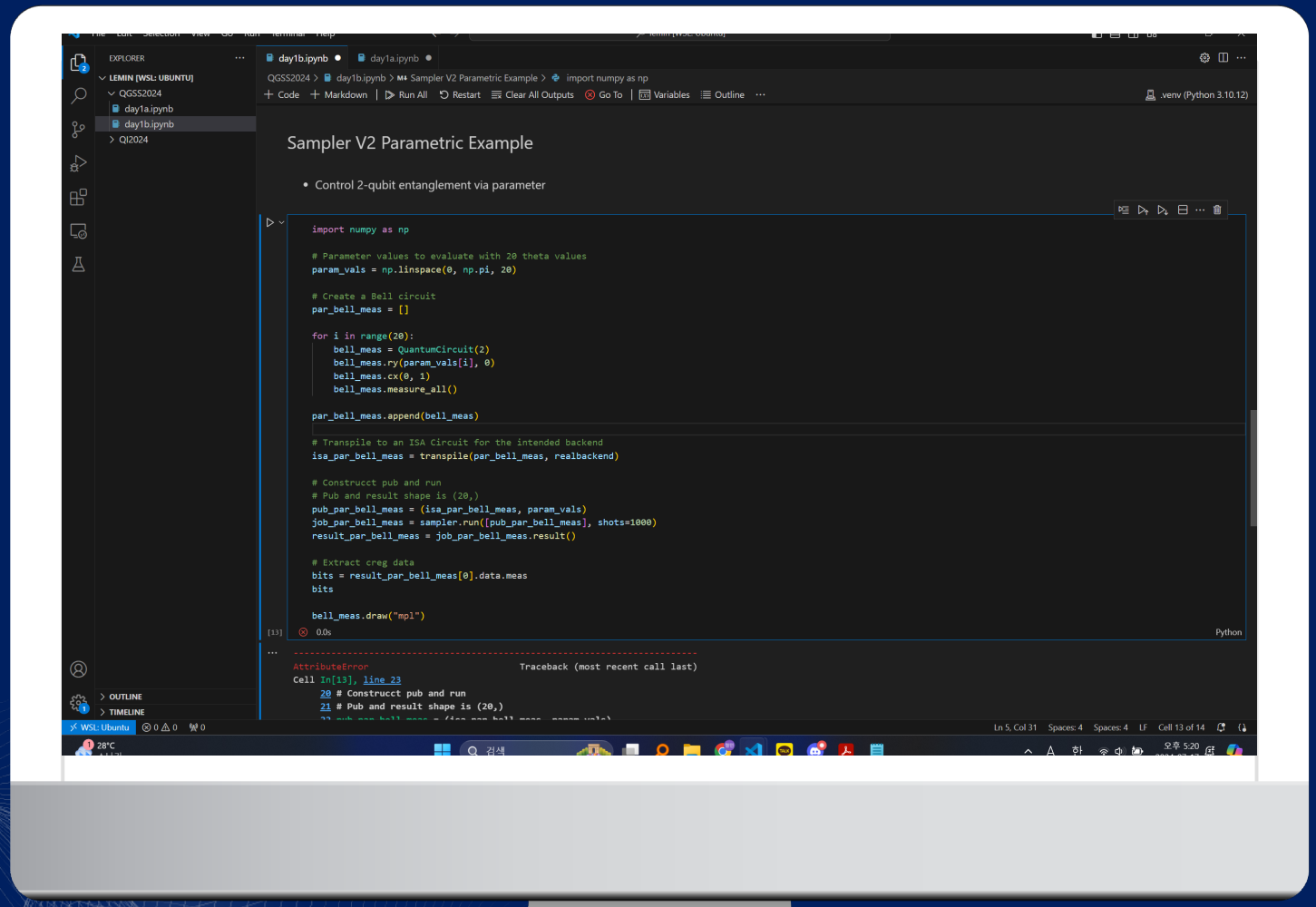
양자 CNN의 구현상의 난제

1. 양자 게이트의 노이즈와 오류율 문제
2. 양자 큐비트 수와 회로 깊이에 따른 제약
3. 현재 양자 컴퓨팅 하드웨어 한계로 인해,
 양자 CNN의 실용적 활용에는 제한적



Assignment

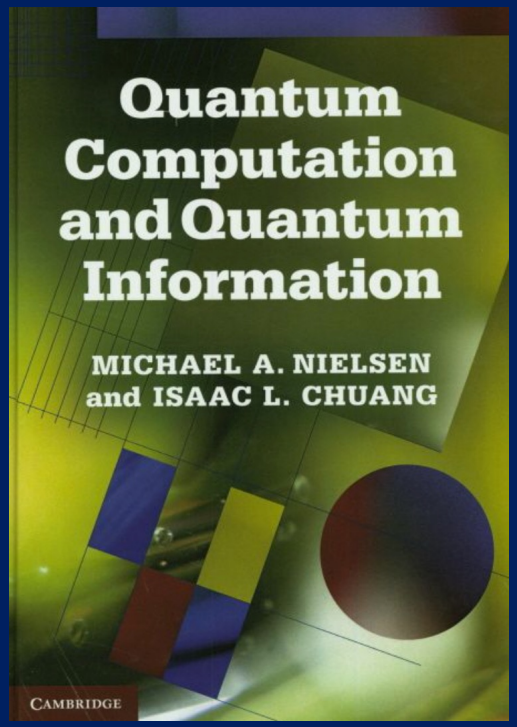
Practice : Coding QML



YONSEI UNIVERSITY

Next?

더 궁금하시다면,
관련 서적들과 논문을 추천드립니다!



YONSEI UNIVERSITY

Thanks!

지금까지 수강해주셔서 감사합니다

Appendix – A, Data encoding – Currently Studied

Hamiltonian encoding

해밀토니안(Hamiltonian)을 데이터에 맞춰 변환하여 표현하는 방식

Methods of Hamiltonian encoding

1. Modeling Hamiltonian : 문제를 해밀토니안으로 변환하여 표현

2. Mapping Qubits: Jordan-Wigner Tr. VS Bravyi-Kitaev T.
해밀토니안을 양자 컴퓨터가 처리할 수 있도록 큐비트에 맞춰 변환

3. Time Evolution : Trotter-Suzuki Decomposition
변환된 해밀토니안을 이용해 양자 회로에서 시간 변화를 계산

Appendix – A, Data encoding – Currently Studied

1. Modeling Hamiltonian

관심 있는 물리적 시스템이나 문제를 나타내는 해밀토니안을 정의

2. Mapping Qubits

1. Jordan-Wigner T.
페르미온 연산자를 파울리 연산자(Pauli operator)로 변환하는 과정

$$c_i = \left(\prod_{j=1}^{i-1} Z_j\right)\sigma_i^-, c_i^\dagger = \left(\prod_{j=1}^{i-1} Z_j\right)\sigma_i^+$$

2. Bravyi-Kitaev T.
Jordan-Wigner 변환의 단점을 보완하여 큐비트 수와 연산 효율성을 개선한 방식

Data encoding – Currently Studied

Time Evolution?

특정 시간 동안 시스템의 상태가 어떻게 변하는지 설명, 일반적으로 $U(t) = e^{-iHt}$연산자를 사용한다.

3. Trotter-Suzuki Decomposition

$$e^{-iHt} \approx \prod_{k} e^{-\frac{iH_k \Delta t}{N}}$$

연속적인 t가 아닌 Δt로 잘라서 Time Evolution을 근사적으로 구현하는 방법
이 과정들 통하여 데이터를 인코딩하고 연산할 수 있게 된다.

한계

Qubit개수 대비 polynomial 하게 증가하는 회로들..

Appendix – B, Theorical Background

Basic of Parameter-shift rule

- 양자 기대값의 특성을 잘 생각해 보면, 다음과 같은 방식을 생각해 볼 수 있다.

- $let\ U_i(\theta_i) = e^{i\theta_i G}$
- $f(x;\theta) = \left\langle 0 \left| U_0^{\dagger}(x) U_i^{\dagger}(\theta_i) \hat{B} U_i(\theta_i) U_0(x) \right| 0 \right\rangle = \left\langle x \left| U_i^{\dagger}(\theta_i) \hat{B} U_i(\theta_i) \right| x \right\rangle$
- $f(x;\theta) = \sum_{j,k=1}^{d} x_j^* \left[e^{i\theta_i w_j} \right]^* b_{jk} e^{i\theta_i w_k} x_k$
- $= \sum_{j=1}^{d} \left| x_j \right|^2 b_{jj} + \sum_{j,k=1, j<k}^{d} [x_j^* b_{jk} x_k e^{i\theta_i (w_k - w_j)} + x_j \left[b_{jk} \right]^* x_k^* \left[e^{i\theta_i (w_k - w_j)} \right]^*]$
- $let\ \{\Omega_l\}_{l \in \{R\}} = \{w_k - w_j | j, k \in \{d\}, w_k > w_j\}$
- $c_{jk} = x_j^* b_{jk} x_k, a_0 = \sum_{j=1}^{d} \left| x_j \right|^2 b_{jj}$
- $f(x;\theta) = a_0 + \sum_{l=1}^{R} c_l e^{i\theta_i \Omega_l} + \sum_{l=1}^{R} c_l^* e^{-i\theta_i \Omega_l}\ \ let\ c_l = \frac{1}{2}(a_l - i b_l), a_l, b_l \in R$
- $f(x;\theta) = a_0 + \sum_{l=1}^{R} [a_l \cos(\Omega_l \theta_i) + b_l \sin(\Omega \theta_i)]$: classical function evaluation

Appendix – B, Theorical Background

Parameter-shift rule

- 양자 회로에서는 파라미터에 대한 미분을 고전적으로 계산하기 어렵다.
- 따라서 파라미터의 특정 두 가지 변화값을 사용하여 미분을 계산하는 방식으로 연산
- $let\ U_i(\theta_i) = e^{i\theta_i G}$
- $f(x;\theta) = \left\langle 0 \left| U_0^{\dagger}(x) U_i^{\dagger}(\theta_i) \hat{B} U_i(\theta_i) U_0(x) \right| 0 \right\rangle = \left\langle x \left| U_i^{\dagger}(\theta_i) \hat{B} U_i(\theta_i) \right| x \right\rangle$
- Let
- $f(x;\theta+s) = a_0 + \sum_{l=1}^{R} [a_l \cos(\Omega_l \theta_i + \Omega s) + b_l \sin(\Omega\theta_i + \Omega s)]$
- $f(x;\theta-s) = a_0 + \sum_{l=1}^{R} [a_l \cos(\Omega_l \theta_i - \Omega s) + b_l \sin(\Omega\theta_i - \Omega s)]$
- $f(x;\theta+s) - f(x;\theta-s) = a_1[-2\sin(\Omega\theta_i)\sin(\Omega s)] + b_1[2\cos(\Omega\theta_i)\sin(\Omega s)]$
- $\frac{[f(x;\theta+s)-f(x;\theta-s)]}{2\sin(\Omega s)} = -a_1 \sin(\Omega\theta_i) + b_1 \cos(\Omega\theta_i)$
- $then\ \nabla_{\theta_i} f(x;\theta) = \frac{[f(x;\theta+s)-f(x;\theta-s)]}{2\sin(\Omega s)} \Omega$: 미분을 양자회로 내 연산