日本語訳『Qiskit Textbook』勉強会

第4章 アプリケーションのための量子アルゴリズム

4.1.4 Groverのアルゴリズムを用いた充足可能性問題の解法
4.1.5 PyTorchとQiskitを用いた量子古典ハイブリッド・ニューラル・ネットワーク

Kaori Namba

Senior Software Engineer

4.1.4
Groverのアルゴリズムを用いた充足可能性問題の解法

充足可能性問題

～ Satisfiability problem, SAT problem ～

- 一つの命題論理式が与えられたとき、それに含まれる変数の値を真か偽にうまく定めることによって、全体の値を‘真’にできるか (by Wikipedia)
- ‘真’となるとき「充足している」という
- 充足解を探す探索問題
- 記号
  - ¬：論理否定 , ∨：論理和, ∧：論理積

3-SAT問題

変数が3つの充足可能性問題

例

$$f(v_1, v_2, v_3) = (\neg v_1 \vee \neg v_2 \vee \neg v_3) \wedge (v_1 \vee \neg v_2 \vee v_3) \wedge (v_1 \vee v_2 \vee \neg v_3) \wedge (v_1 \vee \neg v_2 \vee \neg v_3) \wedge (\neg v_1 \vee v_2 \vee v_3)$$

は、

$$(v_1, v_2, v_3) = (0,0,0)\ or\ (1,0,1)\ or\ (1,1,0)$$

の充足解を持つ

真理値表

v_1	v_2	v_3	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

DIMACS CNF - SAT問題の入力

- c から始まる行はコメント
- 最初の非コメント行は、p cnf nbvar nbclauses
  - cnf：入力がCNF形式
  - nbvar：変数の数
  - nbclauses：節の数
- その後の行は各節に対応
  - -nbvar から nbvarの間の個別の非Null値の数列で、行は0で終わる
  - i番目の変数が、否定の場合は-i, そうでない場合はiと記載する
  - i と -i を同時に含むことはできない
- 例題：$f(v_1, v_2, v_3) = (\neg v_1 \vee \neg v_2 \vee \neg v_3) \wedge (v_1 \vee \neg v_2 \vee v_3) \wedge (v_1 \vee v_2 \vee \neg v_3) \wedge (v_1 \vee \neg v_2 \vee \neg v_3) \wedge (\neg v_1 \vee v_2 \vee v_3)$

例題のDIMACS CNF

```
c example DIMACS CNF 3-SAT
p cnf 3 5
-1 -2 -3 0
1 -2 3 0
1 2 -3 0
1 -2 -3 0
-1 2 3 0
```

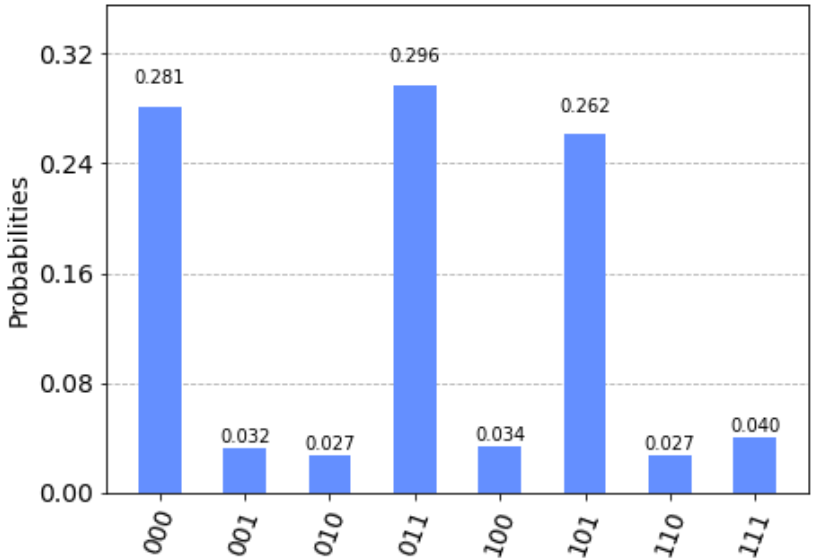

DIMACS：The Center for Discrete Mathematics & Theoretical Science
CNF: Conjunctive normal form (連言標準形)

LogicalExpressionOracleクラス

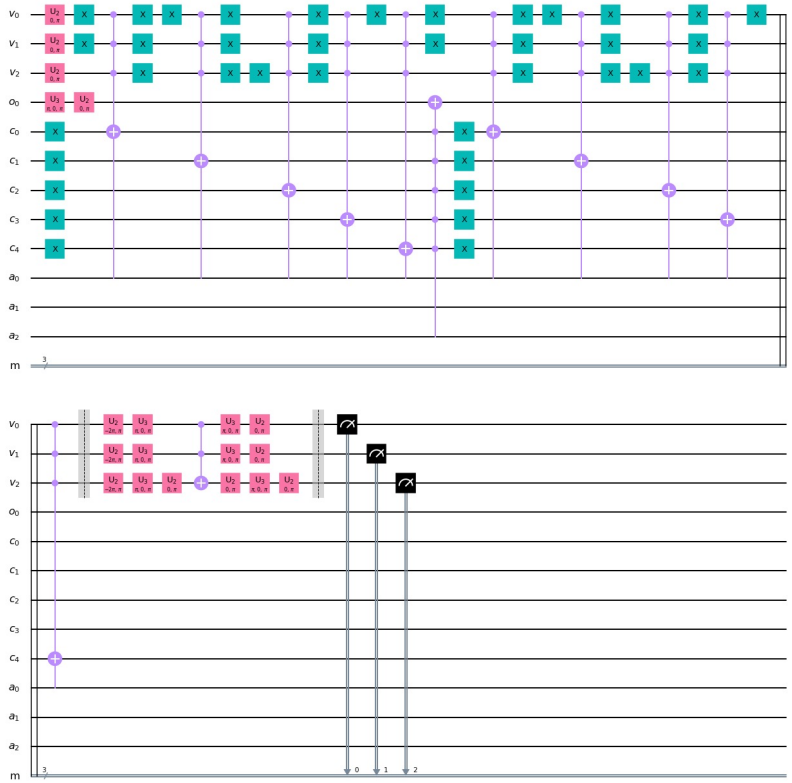
- DIMACS CNF構文を解析し、対応するOracle回路の構築

```
input_3sat = '''
c example DIMACS-CNF 3-SAT
p cnf 3 5
-1 -2 -3 0
1 -2 3 0
1 2 -3 0
1 -2 -3 0
-1 2 3 0
'''
oracle = LogicalExpressionOracle(input_3sat)
grover = Grover(oracle)
```

量子回路の作成完了！



現実の壁



Transpile

gates
- CX : 436
- U2 : 171
- U1 : 77
- U3 : 45
- Measure : 3
- Barrier : 2

depth = 446

演習問題

$$f(x_1, x_2, x_3) = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$$

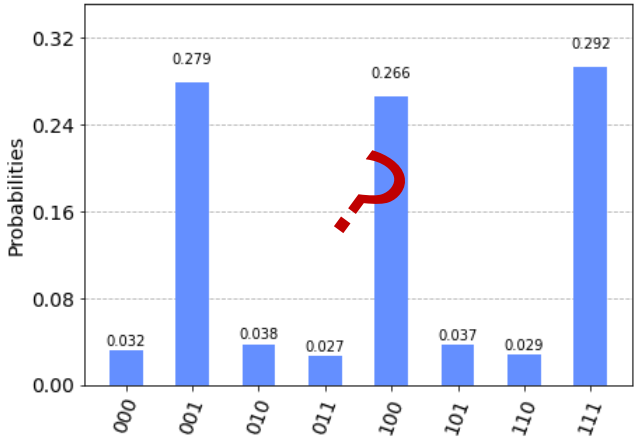
真理値表

x_1	x_2	x_3	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

コード

```
input_3sat = '''
c practice DIMACS-CNF 3-SAT
p cnf 3 3
1 2 -3 0
-1 -2 -3 0
-1 2 3 0

'''

oracle =
LogicalExpressionOracle(input_3sat)

grover = Grover(oracle)
```

シミュレーターの結果



演習問題 – 改

$$f(x_1, x_2, x_3) = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$$

真理値表

x_1	x_2	x_3	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

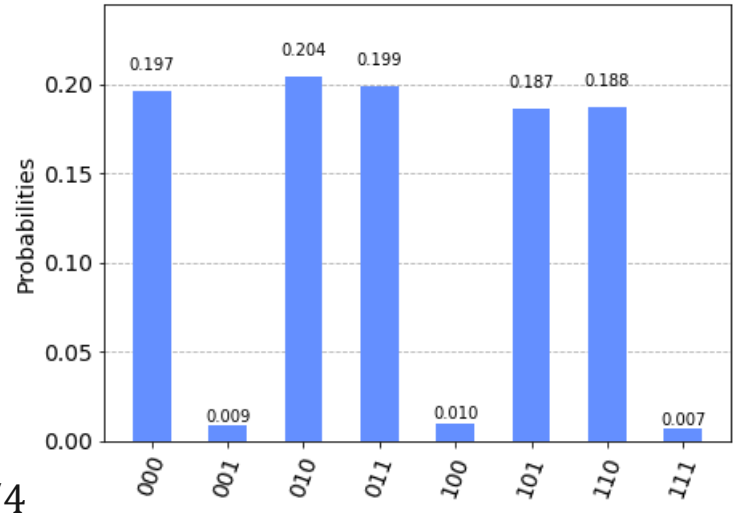
コード

```
input_3sat = '''
c practice DIMACS-CNF 3-SAT
p cnf 3 3
1 2 -3 0
-1 -2 -3 0
-1 2 3 0

'''

oracle =
LogicalExpressionOracle(input_3sat)

grover = Grover(oracle,
num_iterations=2)
```

イテレーション数 $= \pi(N/k)^{1/2}/4$

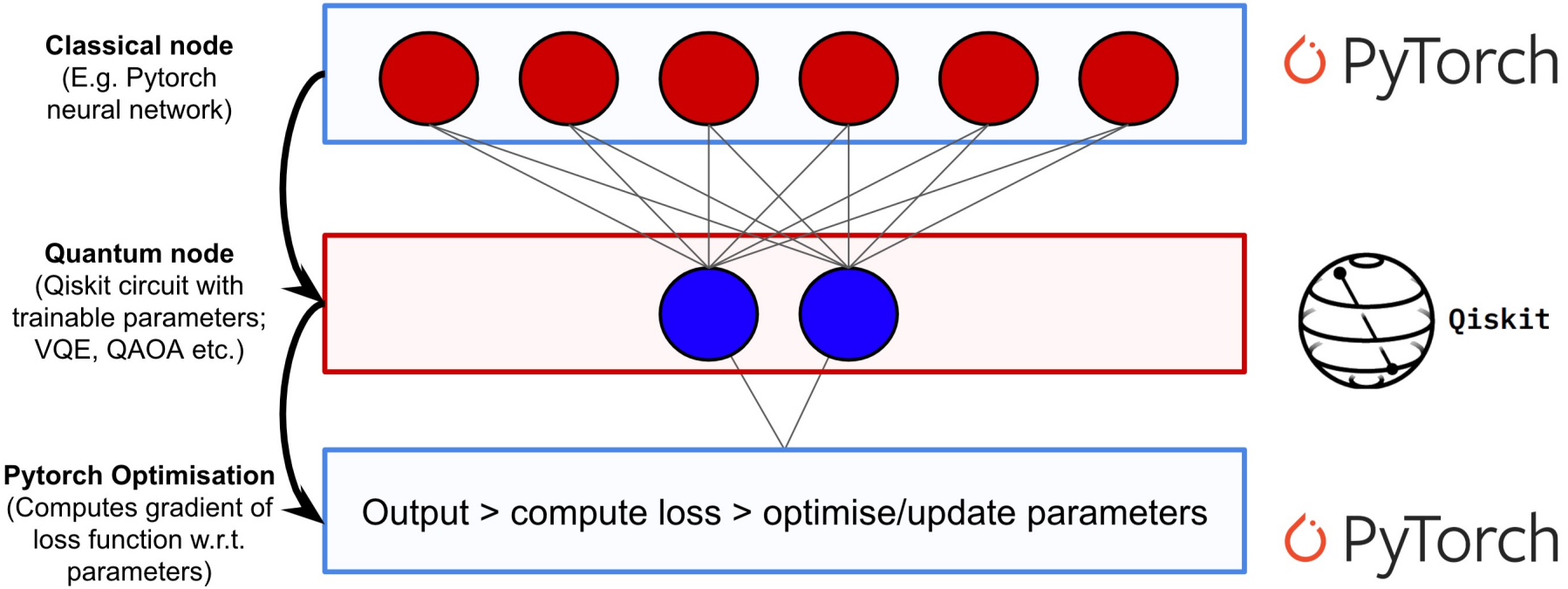
?

シミュレーターの結果



4.1.5
PyTorchとQiskitを用いた量子古典ハイブリッド・ニューラル・ネットワーク

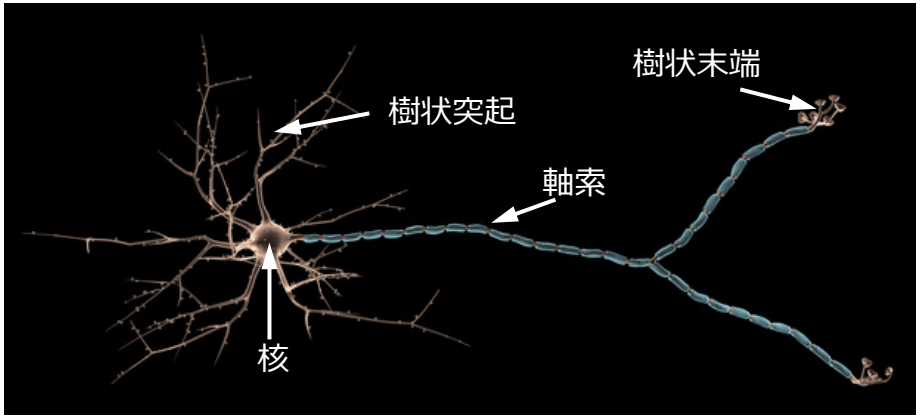
量子古典ハイブリッド・ニューラル・ネットワーク

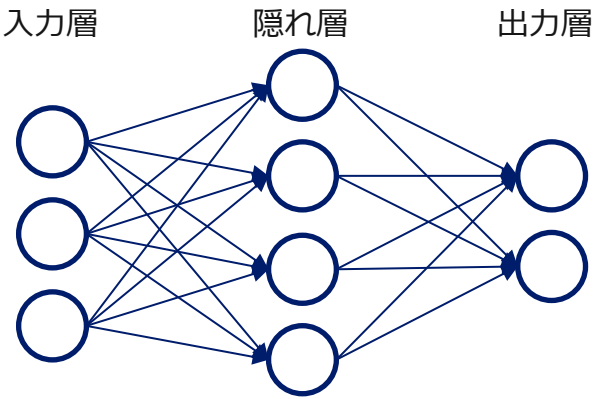


VQE: Variational Quantum Eigensolver
QAOA: Quantum Approximate Optimization Algorithm

ニューラル・ネットワーク

- ヒトのニューロンを模したシステム
  - ニューロン → ノード
  - 結合強度 → 重み
- 入力層に入れられたデータをもとに、出力層の数だけ分類する
- 学習で、重みやバイアスを決める
- フィードフォワード・ニューラル・ネットワーク





今回のハイブリッド・ニューラル・ネットワーク

目標：手書き文字を分類するようなハイブリッド・ニューラル・ネットワークを構築

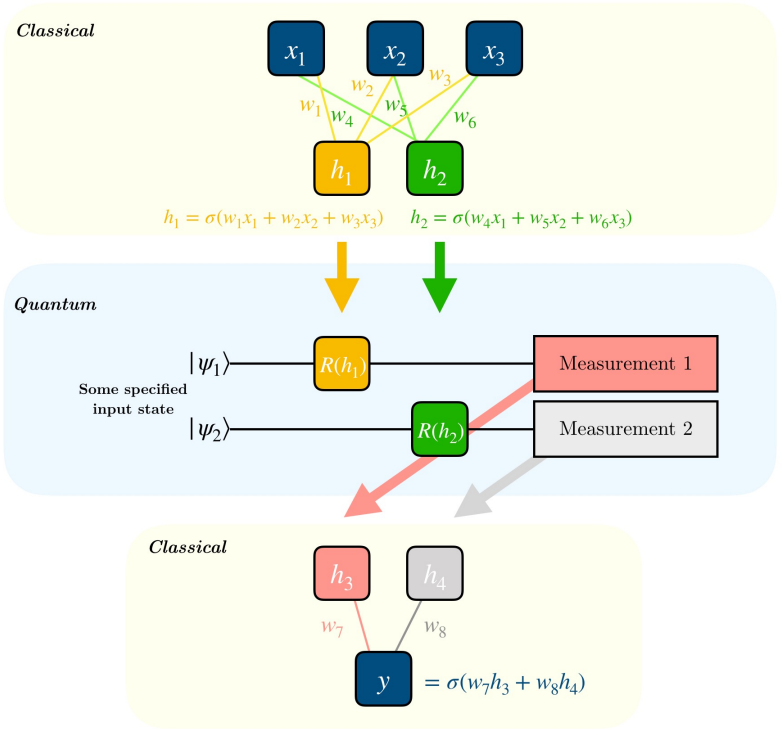
ニューラル・ネットワークの隠れ層を「パラメータ化された量子回路」で構築

- 各ゲートの回転角が古典入力ベクトルの成分によって指定される量子回路
- 量子回路の測定統計が続くの層の入力となる

簡単な例 -> 右図

- σ: 非線形関数
- h_i: 各隠れ層におけるニューロン i の値
- $R(h_i)$: 角度h_iの回転ゲート
- y：最終出力

誤差逆伝播法で学習させる



勾配の計算

パラメーター・シフト・ルール

– 量子回路をブラック・ボックスとみなすとき、パラメーターに対するこのブラック・ボックスの勾配：

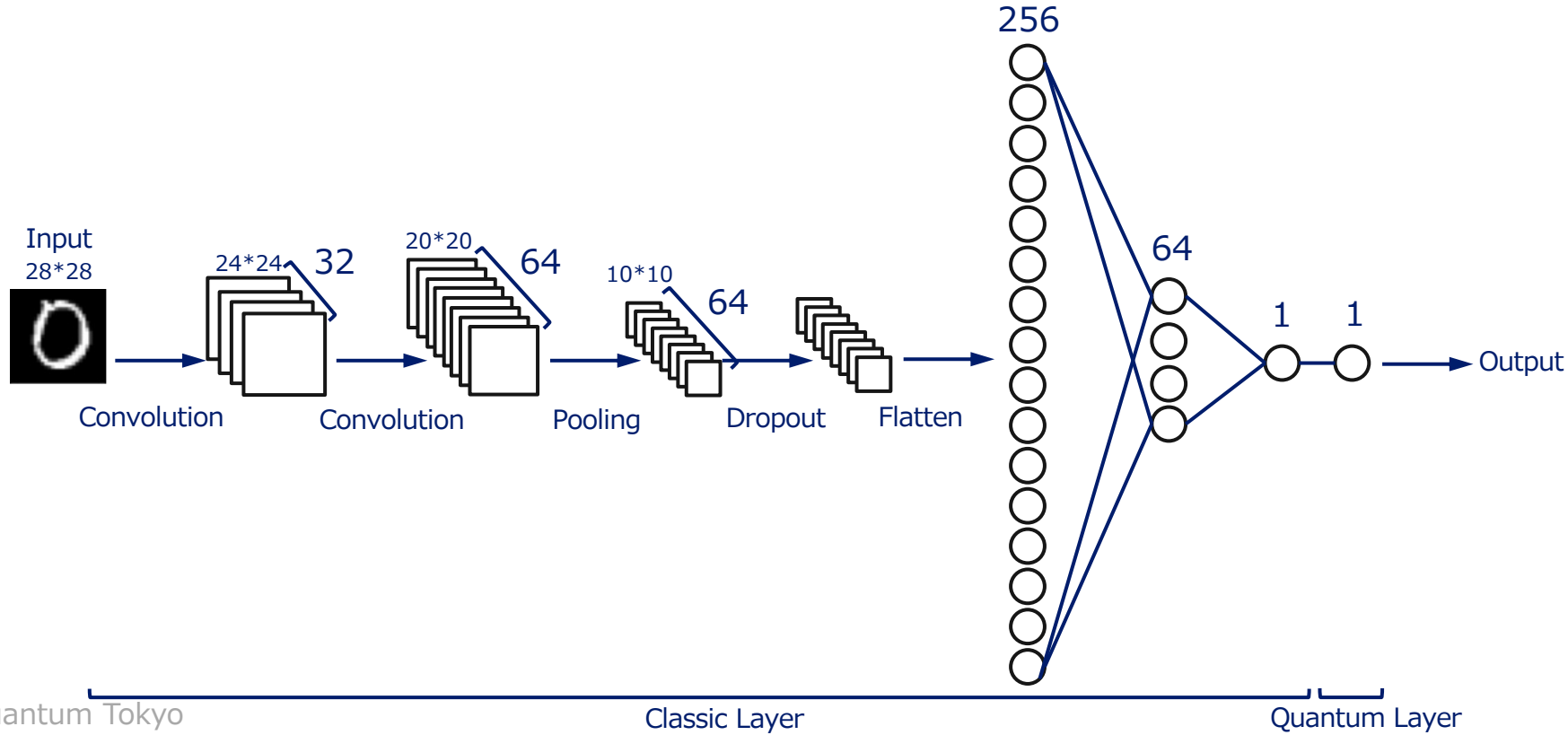
$$\nabla_\theta \boxed{\text{Quantum Circuit}\,(\theta)} = \boxed{\text{Quantum Circuit}\,(\theta + s)} - \boxed{\text{Quantum Circuit}\,(\theta - s)}$$

- θ：量子回路のパラメーター
- s：巨視的シフト
- つまり、回路を $\theta+s$ と $\theta-s$ で評価した時の差

参考：https://arxiv.org/pdf/1905.13311.pdf

目標

MNISTの0と1を判別するハイブリッドネットワークの構築



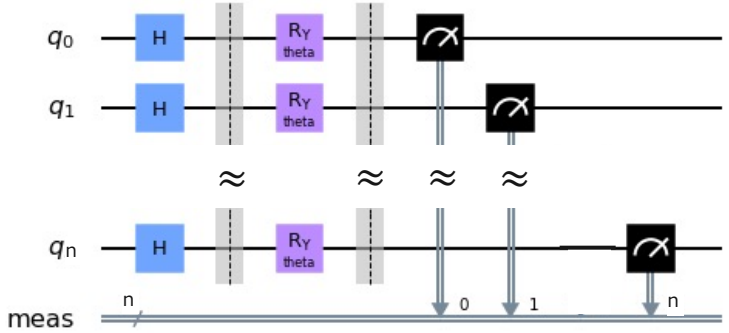
PyTorchにおけるカスタム・レイヤーの作成

- nn.Module を継承したクラスを定義
- Function サブクラスの実装
  - forward() – 結果を計算する関数
  - backward() – 勾配を計算する関数

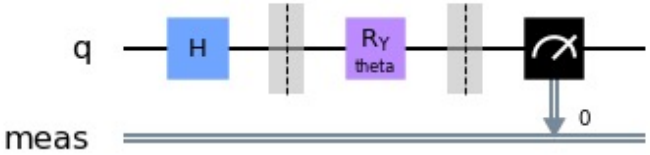
量子回路クラス

- コンストラクタ
  - 角度θのRY回転ゲートを持つn量子ビットの回路の構築
  - 入力：量子ビット数、実行バックエンド、Shot数
- run メソッド
  - 回路を実行し結果を返す
  - 入力：回転角
  - 出力：期待値 $\sigma_z = \sum_i z_i p(z_i)$

量子回路クラス



今回使用する回路（n=1）



ハイブリッド・ニューラル・ネットワークの恩恵とは何か？

- 今回作成したネットワークは古典的なものの方がより良く学習できる！
- 学習された量子層は、エンタングルメントを全く生成しない
  - 古典的にシミュレーション可能
- 量子優位性を実現したい場合は、より洗練された量子層を組み込む必要性あり
- このセクションのポイント
  - MLと量子コンピューティングの技術の統合を考えること
  - PyTorchとQiskitがあれば、楽に考えられる！

Thank you

Kaori Namba
Senior Software Engineer