京大基研量子情報スクール用ノート

Kosuke Mitarai

2019/3/20

1　はじめに

量子機械学習は、機械学習に量子コンピュータのパワーを使おういう試みである。近年の機械学習の発展に伴って、量子アルゴリズムによってそれらを加速しようという動きが加速している。[1] が良いレビュー。このノートでは、サブルーチンとして使われているテクニックやアルゴリズムを一通り見た後、最近の機械学習アルゴリズムについてまとめる。

2　機械学習とは

ここでは教師あり機械学習を例として機械学習とは何かを説明しよう。教師あり機械学習では、以下のデータが訓練データとして与えられる。

- N 個の説明変数: $\{\boldsymbol{x}_i\}$ $(i = 1, \cdots, N)$
  - 例えば、手書き数字画像のピクセル値
- 対応する N 個の教師データ: $\{y_i\}$ $(i = 1, \cdots, N)$
  - 例えば、手書き数字のラベル

教師データ $\{y_i\}$ は、何らかの形で説明変数 $\{\boldsymbol{x}_i\}$ と関連づいていると考えられる。背後に存在するこの「関係」を仮に g としよう。$g(\boldsymbol{x}_i)$ が教師データ y_i を与えると考えるのだ。**機械学習の目的は、与えられた訓練データから g を近似する関数 f を構築することである。** f をデータから構築することを**学習**と呼び、学習によって構築される関数 f を**モデル**と呼ぶ。

通常モデル f には何らかの形のパラメータ付き関数を仮定し、パラメータを調整することによって学習を行う。 以下に少し例を挙げよう。

- 線型回帰 (連続値予測) においては

$$f_{\boldsymbol{w},b}(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} + b \tag{1}$$

- ロジスティック回帰 (2 値分類タスク) においては

$$f_{\boldsymbol{w},b}(\boldsymbol{x}) = \frac{1}{1+e^{\boldsymbol{w}\cdot\boldsymbol{x}+b}} \tag{2}$$

- ニューラルネットでは

$$f_{\{W_l\}}(\boldsymbol{x}) = h_L(W_L h_{L-1}(\cdots W_3 h_2(W_2 h_1(W_1 \boldsymbol{x})))) \tag{3}$$

ここで W_l は行列であり、h_l は何らかの非線形関数である。

これらのモデルでは、$\boldsymbol{w}, b$ や W といったパラメータを調整することで、背後に存在する関係 g を近似する。

3 Quantum arithmetics

Long-term の量子コンピュータを用いた計算では、量子コンピュータ上で適当な古典計算を行うことが多い。量子回路を用いた通常の論理演算は quantum arithmetics と呼ばれている。通常の論理演算をどのように行うか、その一番単純な説明を以下に述べよう。

まず前提として、NAND 演算が可能であればどのような論理演算でもその組み合わせによって実現可能である。NAND 演算は AND + NOT であるが、NOT 演算は X ゲートをかければよい。そこで AND 演算を実現できればユニバーサルな論理演算が可能となるが、それには Toffoli ゲート (controlled bit が 2 つの CNOT ゲート) を用いて、

$$|b_0\rangle |b_1\rangle |0\rangle \xrightarrow{\text{Toffoli}} |b_0\rangle |b_1\rangle |b_0 \cdot b_1\rangle \tag{4}$$

のようにすればよい。したがって原理的には、 Toffoli ゲートと X ゲートをうまく組み合わせることでどのような古典計算も実現できる。しかしここに述べたようなナイーブな実装では必要な補助ビットの数が多くなってしまうため、このビット数を落とすために様々な研究が行われている。

さて、量子コンピュータ上でどのような古典計算もできることを使うと、次のような複雑な演算も可能であることになる。例えば、適当な実数を r、それを表すビット列[*1]を $b(r)$、$f : \mathbb{R} \to \mathbb{R}$ を関数とし、

$$|b(r)\rangle |0^m\rangle |0^n\rangle \to |b(r)\rangle |\text{Gabbage}\rangle |b(f(r))\rangle \tag{5}$$

という演算は、f が古典コンピュータ上で効率的に計算可能な関数ならば、効率的にできる。ここで用意した m qubit のレジスターは計算時の補助ビットとして扱われる部分である。もちろん、様々な実数の重ね合わせ状態を入力すれば

$$\frac{1}{\sqrt{N}} \sum_{i=1}^{N} |b(r_i)\rangle |0^m\rangle |0^n\rangle \to \frac{1}{\sqrt{N}} \sum_{i=1}^{N} |b(r_i)\rangle |\text{Gabbage}\rangle |b(f(r_i))\rangle \tag{6}$$

というある意味での並列計算ができる。

[*1] $b(r)$ は例えば r の浮動小数点数表示

4　データの取り扱い

一般的な機械学習では古典的なデータ (画像など) を取り扱う以上、量子状態に何らかの方法でデータを送り込む必要がある。ここではよく扱われるデータアクセスモデルについて記述する。

4.1　Quantum random access memory

Quantum Random Access Memory (QRAM) とは、あるバイナリデータ x_i のアドレス i を記述する量子ビット列 $|i\rangle$ が与えられたとき、そのデータ x_i を量子ビット列 $|x_i\rangle$ として取り出す機能のことである。具体的には、次のような機能が QRAM と呼ばれることが多いようだ。

$$|i\rangle|0\rangle \xrightarrow{QRAM} |i\rangle|x_i\rangle \tag{7}$$

通常 QRAM はユニタリーなプロセスとして実現することが仮定される。したがって、QRAM の存在を仮定するとき、以下のように重ね合わせ状態を入力すれば、同時並列的にデータを量子状態にエンコードすることが可能である。

$$\frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle|0\rangle \xrightarrow{QRAM} \frac{1}{\sqrt{N}}\sum_{i}|i\rangle|x_i\rangle \tag{8}$$

現実にこれが可能であるかどうかはまだわからない。具体的な実装方法としては [3] などで提案がある。

4.2　振幅エンコーディング

HHL アルゴリズムなど一部のアルゴリズムでは、N 次元のベクトル $\boldsymbol{v}$ を以下のように量子ビットに埋め込んだ状態を用いる。[*2]

$$|\boldsymbol{v}\rangle = \frac{1}{\|\boldsymbol{v}\|}\sum_{i} v_i|i\rangle \tag{9}$$

ただし $\|\boldsymbol{v}\|^2 = \sum_i v_i^2$。この状態を準備する方法には、QRAM を用いる。以下でそのやり方を紹介しよう。なお、以下では簡単のため $\boldsymbol{v}$ は実数であると仮定する。

4.2.1　ナイーブな方法

1. QRAM を用いて $\boldsymbol{v}$ の各成分のビット列表示 $b(v_i)$ をエンコードした以下の状態を用意する。ビット列は n bit であるとする。

$$\frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle|0^n\rangle \xrightarrow{\text{QRAM}} \frac{1}{\sqrt{N}}\sum_{i}|i\rangle|b(v_i)\rangle \tag{10}$$

[*2] このノートではケットの中に太字を書いたとき、それは式 (9) のような状態を表すことにする。逆に太字でないときは、それはすべてビット列を表す。

2. もう一つ m bit のレジスターを用意し、quantum arithmetics によって m bit の精度で $\frac{1}{\pi}\cos^{-1}(v_i)$ を計算する。

$$\frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle\,|b(v_i)\rangle\,|0^m\rangle \xrightarrow{\text{arithmetics}} \frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle\,|b(v_i)\rangle\left|b\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)\right\rangle \tag{11}$$

簡単のため arithmetics に必要な補助ビットは省略しているが、本来はたくさんの補助ビットがあることは心に留めておこう。ここでの $b\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)$ は浮動小数点表示ではなくて、

$$\frac{2}{\pi}\cos^{-1}(v_i) \approx \sum_{k=0}^{m-1} 2^{-k-1} b_k\left(\frac{2}{\pi}\cos^{-1}(v_i)\right) \tag{12}$$

となるようなビット列であるとする。

3. 補助ビットを 1 bit 用意し、$b_k\left(\frac{2}{\pi}\cos^{-1}(v_i)\right)$ を格納しているレジスターから、controlled-R_y ゲートをかける。それぞれのビット b_k が 1 ならば $R_y(2^{-k})$ を補助ビットにかけるという操作をすれば、以下の演算が実現される。

$$\begin{aligned}
&\frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle\,|b(v_i)\rangle\left|b\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)\right\rangle|0\rangle \\
&\xrightarrow{\text{controlled-}R_y} \frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle\,|b(v_i)\rangle\left|b\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)\right\rangle\prod_{k=0}^{m-1}R_y\left(b_k\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)2^{-k}\right)|0\rangle \\
&= \frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle\,|b(v_i)\rangle\left|b\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)\right\rangle R_y\left(\sum_{k=0}^{m-1}b_k\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)2^{-k}\right)|0\rangle
\end{aligned} \tag{13}$$

ここで

$$R_y(\theta)\,|0\rangle = \cos\frac{\theta}{2}\,|0\rangle + \sin\frac{\theta}{2}\,|1\rangle \tag{14}$$

なので、式 (12) から

$$R_y\left(\sum_{k=0}^{m-1}b_k\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)2^{-k}\right)|0\rangle = v_i\,|0\rangle + \sqrt{1-v_i^2}\,|1\rangle \tag{15}$$

である。したがってこのステップによって生成される状態 (式 (13)) は、

$$\frac{1}{\sqrt{N}}\sum_{i=1}^{N}|i\rangle\,|b(v_i)\rangle\left|b\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)\right\rangle\left(v_i\,|0\rangle + \sqrt{1-v_i^2}\,|1\rangle\right) \tag{16}$$

である。

4. 次に、前ステップで R_y をかけた補助ビットを測定する。もし $|0\rangle$ を観測することができれば、量子状態は次のものに収縮する。

$$\frac{1}{\|\boldsymbol{v}\|}\sum_{i=1}^{N}v_i\,|i\rangle\,|b(v_i)\rangle\left|b\left(\frac{1}{\pi}\cos^{-1}(v_i)\right)\right\rangle|0\rangle \tag{17}$$

この状態が得られる確率は $\sum_i v_i^2/N$ である。

5. ステップ 1, 2 に相当するユニタリーの逆演算を行い、いらないレジスターを 0 に戻す。[*3]この操作を終えれば、所望の状態 $|\boldsymbol{v}\rangle = \frac{1}{\|\boldsymbol{v}\|}\sum_i v_i |i\rangle$ を得られる。

なお、v_i が index i から効率的に計算できる関数であるときには、直接 $\cos^{-1}(v_i)$ を計算すればよいので、QRAM を使わなくても $|\boldsymbol{v}\rangle$ を作り出せる。

ここで紹介した方法には、(QRAM を使っていることのほかに) 二つの問題点がある。

- ユニタリー操作でないこと。結果として、$2|\boldsymbol{v}\rangle\langle\boldsymbol{v}| - I$ のような演算子[*4]を作ることができないこと。
- $\sum_i v_i^2$ (データの 2 次モーメント) が $O(N)$ でなければ、成功確率がどんどんと小さくなってしまうこと。

この二点を解決する手法が Prakash [7] により提案された。

4.2.2 Prakash 法

以下で説明する手法を便宜上 Prakash 法と呼ぶことにする。Praksh 法は Grover による提案 [4] をベースとしていると思われる。Prakash 法ではまず、図 1 のように、$\|\boldsymbol{v}\|^2$ を次々に二分割していくデータ構造を準備する。$\|\boldsymbol{v}\|^2$ は規格化しておき、$\|\boldsymbol{v}\|^2 = 1$ となっていることにする。それぞれのノードに保持されているデータには

$$\begin{aligned}&|\text{parent node address}\rangle\,|0^m\rangle\\ &\to |\text{parent node address}\rangle\,|\text{bit-string stored at the childdren node}\rangle\end{aligned} \tag{18}$$

のように $O(\mathrm{poly}\log N)$ 時間でアクセス可能であると仮定する。

さて、簡単な例として、Prakash 法によって図 1 のデータ構造を振幅エンコーディングする方法をいかに記そう。

1. $|000\rangle\,|0^n\rangle$ を準備する。
2. root の子である node 0 のデータを取得して、$|000\rangle\left|b\left(\sum_{i=0}^{3} v_i^2\right)\right\rangle$ とする。
3. 1 ビット目に、4.2.1 と同様の手順で controlled rotation をかけ、node 0 のデータは uncompute する。得られる状態は

$$\left(\sqrt{\sum_{i=0}^{3} v_i^2}\,|0\rangle + \sqrt{\sum_{i=4}^{7} v_i^2}\,|1\rangle\right)|00\rangle\,|0^n\rangle \tag{19}$$

4. 1 ビット目をアドレスビットと見立てて、node 00, node 10 のデータを取得する。状態は

$$\sqrt{\sum_{i=0}^{3} v_i^2}\,|0\rangle\,|00\rangle\left|b\left((v_0^2+v_1^2)/P\right)\right\rangle + \sqrt{\sum_{i=4}^{7} v_i^2}\,|1\rangle\,|00\rangle\left|b\left((v_4^2+v_5^2)/P\right)\right\rangle \tag{20}$$

[*3] このような操作は量子計算で頻出し、**uncompute** と呼ばれる。

[*4] このような演算子はデータを入力とした量子ウォークを実現するために必要子である。

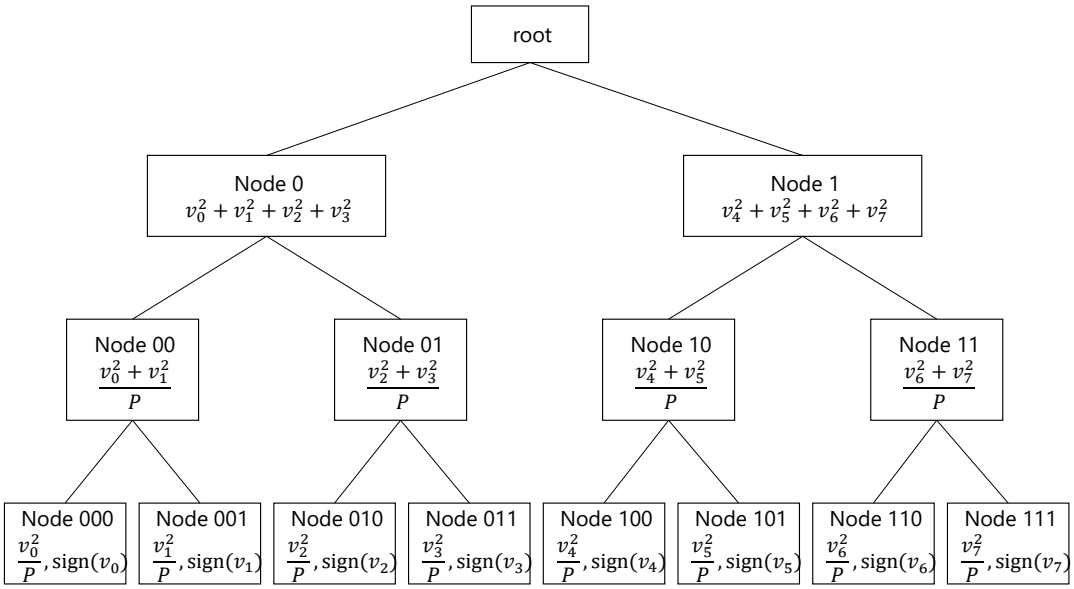

図 1　Prakash [7] による振幅エンコーディングのためのデータ構造。P は各ノードの parent ノードの値。

5. 2 ビット目に同様の手順で controlled rotation をかけ、データビットを uncompute する。得られる状態は

$$\left(\sqrt{v_0^2+v_1^2}\,|00\rangle+\sqrt{v_2^2+v_3^2}\,|01\rangle+\sqrt{v_4^2+v_5^2}\,|10\rangle+\sqrt{v_6^2+v_7^2}\,|10\rangle\right)|0\rangle\,|0^n\rangle \tag{21}$$

6. 最後のデータも同様に取得し、3 ビット目に controlled rotation をかける。最後のノードには v_i の符号も格納されているので、それに応じて符号も処理する。これにより

$$\sum_{i=0}^{7} v_i\,|i\rangle \tag{22}$$

が得られる。

ステップ数は $O(\log N)$ となることは明らかだろう。QRAM 的な機能 (式 (18)) が $O(\mathrm{poly}\log N)$ でできると仮定したので、全体でも $O(\mathrm{poly}\log N)$ の時間で実行可能である。この手法は quantum recommendation system [5] などで使用例がある。ユニタリー操作で完結しているため使いやすい。

5　使用されるアルゴリズム・テクニック

5.1　Swap test

Swap test は量子状態を破壊することなく 2 つの状態 $|\boldsymbol{v}\rangle, |\boldsymbol{w}\rangle$ の overlap $|\langle\boldsymbol{v}|\boldsymbol{w}\rangle|^2$ を測定するアルゴリズムである。図 2 のような量子回路を用いる。補助ビットの Z 期待値をとることで、$|\langle\boldsymbol{v}|\boldsymbol{w}\rangle|^2$ を測定できることは簡単に確かめられる。補助ビットの Z 測定によって $|\boldsymbol{v}\rangle, |\boldsymbol{w}\rangle$ が壊されないことに注意したい。したがって $|\boldsymbol{v}\rangle, |\boldsymbol{w}\rangle$ 一組があれば何回でも測定を繰り返すことが可能である。

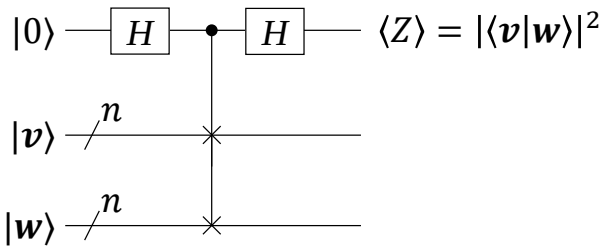

図 2　Swap test

5.2　Density matrix exponentiation

Density matrix exponentiation [6] は、ある密度行列 ρ をハミルトニアンとする時間発展演算子 $\exp(i\rho T)$ を量子状態に作用させるためのテクニックである。

ρ を n-qubit の密度行列、$\exp(i\rho T)$ をかけるターゲットとなる n-qubit の量子状態を σ、S_n を n-qubit 量子状態に対する swap ゲートとする。δ を実数とし、$\exp(-i\delta S_n)$ というゲートを考えよう。$S_n^2 = I$ であるから、

$$\exp(-i\delta S_n) = I\cos\delta - iS_n\sin\delta \tag{23}$$

である。Density matrix exponentiation では、以下の操作を行う。

1. $\rho\otimes\sigma$ を用意し、$\exp(-i\delta S_n)$ を作用させる。

$$\exp(-i\delta S_n)\rho\otimes\sigma\exp(-i\delta S_n) = \rho\otimes\sigma\cos^2\delta + \sigma\otimes\rho\sin^2\delta - i(S_n\rho\otimes\sigma - \rho\otimes\sigma S_n)\cos\delta\sin\delta \tag{24}$$

2. もともと ρ のあったレジスターを無視する。すなわち、部分トレースをとる。常識のそれぞれの項は

$$\rho\otimes\sigma \xrightarrow{\text{partial trace}} \sigma \tag{25}$$

$$\sigma\otimes\rho \xrightarrow{\text{partial trace}} \rho \tag{26}$$

$$\begin{aligned} S_n\rho\otimes\sigma &= S_n\sum_{i,j,k,l}\rho_{ij}\sigma_{kl}\left|i\right\rangle\left\langle j\right|\otimes\left|k\right\rangle\left\langle l\right| \\ &= \sum_{i,j,k,l}\rho_{ij}\sigma_{kl}\left|k\right\rangle\left\langle j\right|\otimes\left|i\right\rangle\left\langle l\right| \\ &\xrightarrow{\text{partial trace}} \sum_{i,j,k,l}\rho_{ij}\sigma_{kl}\delta_{kj}\left|i\right\rangle\left\langle l\right| = \rho\sigma \end{aligned} \tag{27}$$

$$\rho\otimes\sigma S_n \xrightarrow{\text{partial trace}} \sigma\rho \tag{28}$$

となるので、もともと σ のあったレジスターに残る量子状態は

$$\sigma_1 = \sigma\cos^2\delta + \rho\sin^2\delta - i[\rho,\sigma]\cos\delta\sin\delta \tag{29}$$

である。δ を十分小さくとっておき、δ^2 の項を無視することにすると

$$\sigma_1 = \sigma - i[\rho, \sigma]\delta + O(\delta^2) \tag{30}$$

3. ρ のコピーを T/δ 個用意し、ステップ 1, 2 を T/δ 回繰り返す。

アルゴリズムの誤差を評価しよう。ステップ 1 と 2 を合わせた操作を F_ρ とする。

$$F_\rho \sigma = \sigma - i[\rho, \sigma]\delta + O(\delta^2) \tag{31}$$

一方で

$$e^{-i\rho\delta}\sigma e^{i\rho\delta} = \sigma - i[\rho, \sigma]\delta + O(\delta^2) \tag{32}$$

なので、

$$F_\rho \sigma = e^{-i\rho\delta}\sigma e^{i\rho\delta} + O(\delta^2) \tag{33}$$

であるといえる。式 (33) の両辺に F_ρ を繰り返しかければ、

$$F_\rho^{T/\delta} \sigma = e^{-i\rho T}\sigma e^{i\rho T} + O(\delta^2) \tag{34}$$

となり、ステップ幅 δ の 2 乗のオーダーの誤差が現れることがわかる。

5.3 位相推定

量子位相推定 (Quantum Phase Estimation, QPE) は、適当なユニタリー U が与えられたとき、その固有値 $e^{i2\pi\phi_i}$ をビット列として取り出すためのアルゴリズムである。Nielsen-Chuang の教科書などを参照。QPE は、U の固有値 $e^{i2\pi\phi_i}$ に対応する固有ベクトルを $|u_i\rangle$ とするとき、controlled-U と量子フーリエ変換を用いて、次の変換を実現する。

$$\sum_i \alpha_i |u_i\rangle |0^m\rangle \xrightarrow{\text{QPE}} \sum_i \alpha_i |u_i\rangle |b(\phi_i)\rangle \tag{35}$$

ただし $b(\phi_i)$ は $\phi_i = \sum_{k=0}^{m-1} 2^{-k-1} b_k(\phi_i)$ となるビット列である。

5.4 量子ウォーク

N 個の正規直交ベクトル $\{|\boldsymbol{v}_i\rangle\}_{i=1}^N$ と M 個の正規直交ベクトル $\{|\boldsymbol{w}_j\rangle\}_{j=1}^M$ を用いて以下の演算子を定義しよう。

$$W = \left(2\sum_{i=1}^{N} |\boldsymbol{v}_i\rangle \langle \boldsymbol{v}_i| - I\right)\left(2\sum_{j=1}^{M} |\boldsymbol{w}_j\rangle \langle \boldsymbol{w}_j| - I\right) \tag{36}$$

この演算子 W は以下の有用な性質を持つ。D を $D_{ij} = \langle \boldsymbol{v}_i | \boldsymbol{w}_j \rangle$ を成分とする行列とし、D の右特異ベクトルのうち一つを $\boldsymbol{a}$、対応する左特異ベクトルを $\boldsymbol{b}$ 、対応する特異値を $\sigma = \cos\theta \neq 0$ とする。このとき、

- $\{\sum_i a_i |\boldsymbol{v}_i\rangle, \sum_j b_j |\boldsymbol{w}_j\rangle\}$ という 2 つのベクトルによって張られる空間は W の不変部分空間。
- その不変部分空間における W の固有値は $e^{\pm i2\theta}$。
- 対応する固有ベクトルは $|\mu_\pm\rangle = e^{\pm i\theta} \sum_i a_i |\boldsymbol{v}_i\rangle + \sum_j b_j |\boldsymbol{w}_j\rangle$。

証明は説明が込み入るので避ける。$|\boldsymbol{v}_i\rangle, |\boldsymbol{w}\rangle_j$ をうまく作れば、任意の行列 D についてその固有値を間接的に持つ演算子 W を作り出せることに注意したい。

5.5 HHL アルゴリズム

多くの量子機械学習アルゴリズムは、Harrow-Hassidim-Lloyd による逆行列計算アルゴリズム (**HHL アルゴリズム**) をベースとして構築されている。

HHL アルゴリズムは、$N \times N$ の疎行列 A、N 次元のベクトル $\boldsymbol{v}$ について、

$$|\boldsymbol{v}\rangle \xrightarrow{\text{HHL}} \left|A^{-1}\boldsymbol{v}\right\rangle \tag{37}$$

という変換を効率良く計算するアルゴリズムである。その計算量は、現在知られているベストのもので、$O(s\kappa \text{poly}\log(s\kappa/\epsilon))$ である。(s は A の sparsity・κ は A の条件数・ϵ は出力状態の $\left|A^{-1}\boldsymbol{v}\right\rangle$ からの誤差) 一方で古典ベストのアルゴリズムは共役勾配法であり、その実行時間は $O(Ns\kappa \log(1/\epsilon))$ なので、行列のスパース性 ($s = O(\text{poly}\log N)$) を仮定すれば、HHL アルゴリズムは行列の次元 N について指数加速を実現している。

オリジナルバージョンの HHL アルゴリズムは大まかに以下のようなものである。A の固有値を $0 < \lambda_i < 1$、規格化された固有ベクトルを $\boldsymbol{a}_i$ とするとき、

1. 状態 $|\boldsymbol{v}\rangle$ を QRAM などを駆使して用意する。
2. controlled-$e^{i2\pi A}$ を使った位相推定によって $\sum_i \langle \boldsymbol{v}|\boldsymbol{a}_i\rangle |\boldsymbol{a}_i\rangle |\lambda_i\rangle$ をつくる。
3. 4.2.1 と同様のアルゴリズムによって、補助ビットに controlled rotation をかけ

$$\sum_i \langle \boldsymbol{v}|\boldsymbol{a}_i\rangle |\boldsymbol{a}_i\rangle |\lambda_i\rangle \left(\frac{1}{\lambda_i}|0\rangle + \sqrt{1-\frac{1}{\lambda_i^2}}|1\rangle\right) \tag{38}$$

   を得る。
4. 補助ビットを測定して $|0\rangle$ が観測されれば、$\frac{1}{\|A^{-1}\boldsymbol{v}\|}\left|A^{-1}\boldsymbol{v}\right\rangle$ を得る。

次のセクションで、HHL アルゴリズム、もしくはそれと似たアイデアを用いた量子機械学習アルゴリズムについて述べていく。

6 HHL アルゴリズムベースの量子機械学習

6.1 量子線型回帰 [9]

6.1.1 線型回帰

線型回帰では、

$$f_{\boldsymbol{w},b}(\boldsymbol{x}) = \boldsymbol{w}\cdot\boldsymbol{x} + b \tag{39}$$

という形の関数によって教師データからモデル $f_{\boldsymbol{w},b}$ を作成する。データ $\boldsymbol{x}$ は M 次元の実数ベクトル、データ数は N とする。オフセットのパラメータ b は、$\boldsymbol{x} = \{x_i\}_{i=1}^{M}$ に一つデータ x_0 を追加し、常に $x_0 = 1$ としておけば $\boldsymbol{w}$ に含めてしまうことができる。そこで以下では、

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}\cdot\boldsymbol{x} \tag{40}$$

をモデルとする。コスト関数としては二乗誤差

$$L(\boldsymbol{w}) = \sum_{i=1}^{N}\left(f_{\boldsymbol{w}}(\boldsymbol{x}_i) - y_i\right)^2 \tag{41}$$

を選ぶ。$\boldsymbol{x}_i$ の成分は $\boldsymbol{x}_i = \{x_{ij}\}_{j=1}^{M}$ と書く。目標はこのコスト関数 $L(\boldsymbol{w})$ を最小とする $\boldsymbol{w}$ を見つけることである。これは $\frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}} = 0$ の条件によって達成できる。この条件は以下の方程式に帰着する。

$$\sum_{k=1}^{N}\sum_{j=1}^{N} x_{ji}x_{jk}w_k = \sum_{j=1}^{N} x_{ji}y_j \tag{42}$$

X を x_{ij} をその成分として持つ行列とし、その転置を X^T と書くと

$$\boldsymbol{w}^* = (X^TX)^{-1}X\boldsymbol{y} \tag{43}$$

が最適なモデルパラメータである。$(X^TX)^{-1}X \equiv X^+$ は X の疑似逆行列と呼ばれ、X の特異値分解 $X = U\mathrm{diag}(\sigma_1\cdots\sigma_M)V^\dagger$ を用いると以下のように書き下せる。特異値は大きさの降順に並べ、r 個の特異値が非ゼロであったとするとき、

$$X^+ = V\mathrm{diag}\left(\frac{1}{\sigma_1},\cdots,\frac{1}{\sigma_r},0,\cdots,0\right)U^\dagger \tag{44}$$

となる。

未知のデータ $\tilde{\boldsymbol{x}}$ が入力されたとき、モデルによる予測値は、上記で得た最適な $\boldsymbol{w}^*$ を使って $f_{\boldsymbol{w}^*}(\tilde{\boldsymbol{x}}) = \sum_i w_i^*\tilde{x}_i$ によって与えられる。これを書き下すと

$$f_{\boldsymbol{w}^*}(\tilde{\boldsymbol{x}}) = \sum_{i=1}^{r}\frac{1}{\sigma_i}\left(\sum_{j=1}^{M}x_jv_{ji}\right)\left(\sum_{k=1}^{N}u_{ik}y_k\right) \tag{45}$$

ただし v_{ij}, u_{ij} は V, U の (i,j) 成分。 以下では $\boldsymbol{v}_i = \{v_{ji}\}_{j=1}^{M}$、$\boldsymbol{u}_i = \{u_{ji}\}_{j=1}^{N}$ と書くことにする。

6.1.2 量子アルゴリズム

ここでは [9] による HHL アルゴリズムをベースとした量子アルゴリズムについて述べる。先に概要を以下に書いておく。

1. X^TX という行列を密度行列として準備する。つまり $\rho_{X^TX} = \sum_{i,j}(X^TX)_{ij}\,|i\rangle\langle j|$ を作る。
2. Density matrix exponentiation によって $\exp(-i2\pi\rho_{X^TX})$ の位相推定を状態 $\frac{1}{\sqrt{N}}\sum_{i=1}^{N}|\boldsymbol{x}_i\rangle\,|i\rangle$ に対して実行し、σ_i^2 のビット列を取得する。
3. σ_i^2 の逆数を補助ビットへの controlled rotation によって乗じる。すると $\sum_{i=1}^{r}\sigma_r^{-1}\,|\boldsymbol{v}_r\rangle\,|\boldsymbol{u}_r\rangle$ が得られる。
4. $|\tilde{\boldsymbol{x}}\rangle\,|\boldsymbol{y}\rangle$ という状態を作り、swap test によって内積を評価する。

わかりにくいと思われる点について以降で補足説明していく。

まず密度行列の準備について。実は、データを振幅エンコードした状態 $\sum_{i=1}^{N}|\boldsymbol{x}_i\rangle\,|i\rangle$ の部分系が X^TX をエンコードした密度行列を持っている。$|i\rangle$ の部分について部分トレースをとると、

$$\begin{aligned}\left(\sum_{i=1}^{N}|\boldsymbol{x}_i\rangle\,|i\rangle\right)\left(\sum_{j=1}^{N}\langle\boldsymbol{x}_j|\,\langle j|\right) &= \sum_{i=1}^{N}\sum_{j=1}^{N}|\boldsymbol{x}_i\rangle\langle\boldsymbol{x}_j|\otimes|i\rangle\langle j| \\ &\xrightarrow{\text{partial trace}} \sum_{i=1}^{N}\sum_{j=1}^{N}|\boldsymbol{x}_i\rangle\langle\boldsymbol{x}_j|\,\delta_{ij} \\ &= \sum_{i=1}^{N}|\boldsymbol{x}_i\rangle\langle\boldsymbol{x}_i| \\ &= \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{M}x_{ij}x_{ik}\,|j\rangle\langle k| \end{aligned} \tag{46}$$

となって ρ_{X^TX} に等しいことがわかる。

位相推定の部分。まず、$\sum_{i=1}^{N}|\boldsymbol{x}_i\rangle\,|i\rangle$ を Schmidt 分解すると、

$$\begin{aligned}\sum_{i=1}^{N}|\boldsymbol{x}_i\rangle\,|i\rangle &= \sum_{i=1}^{N}\sum_{j=1}^{M}x_{ij}\,|j\rangle\,|i\rangle \\ &= \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{k=1}^{r}u_{ik}\sigma_k v_{jk}\,|j\rangle\,|i\rangle \\ &= \sum_{k=1}^{r}\sigma_k\,|\boldsymbol{v}_k\rangle\,|\boldsymbol{u}_k\rangle \end{aligned} \tag{47}$$

$\exp(i2\pi\rho_{X^TX})$ を作用させるのはこのうち $|\boldsymbol{v}_i\rangle$ の部分。つまり、$|\boldsymbol{v}_i\rangle$ の部分と $\sum_{i=1}^{N}|\boldsymbol{x}_i\rangle\,|i\rangle$ の $|i\rangle$ の部分に対して少しだけ swap ゲートをかけることを繰り返し、5.2 の density matrix

exponentiation を行う。$\boldsymbol{v}_k$ は X の右特異ベクトルなので、X^TX の固有ベクトルであり、その固有値は σ_i^2 である。したがって位相推定をかけると以下の状態が得られる。

$$\sum_{k=1}^{r} \sigma_k \left|\boldsymbol{v}_k\right\rangle \left|\boldsymbol{u}_k\right\rangle \left|b(\sigma_k^2)\right\rangle \tag{48}$$

さらに、HHL アルゴリズムと同様に controlled rotation → 測定 → 位相推定の uncompute を行うことによって

$$\frac{1}{\sqrt{\sum_k \sigma_k^{-2}}} \sum_{k=1}^{r} \frac{1}{\sigma_k} \left|\boldsymbol{v}_k\right\rangle \left|\boldsymbol{u}_k\right\rangle \tag{49}$$

までもっていくことができる。これで式 (45) の v_{ji} と u_{ik} の部分を構築することができた。

最後に、式 (49) の状態を用いて未知データ $\tilde{\boldsymbol{x}}$ に関する予測を行う。式 (45) から、式 (49) の状態と $\left|\boldsymbol{x}\right\rangle \left|\boldsymbol{y}\right\rangle$ という状態との内積を評価できれば良いことがわかる。内積を一番簡単に評価できるのは 5.1 で紹介した swap test だが、$\left|\tilde{\boldsymbol{x}}\right\rangle \left|\boldsymbol{y}\right\rangle$ と $\frac{1}{\sqrt{\sum_k \sigma_k^{-2}}} \sum_{k=1}^{r} \frac{1}{\sigma_k} \left|\boldsymbol{v}_k\right\rangle \left|\boldsymbol{u}_k\right\rangle$ に対する swap テストをただ行うだけでは、符号の情報が失われてしまう。そこで解決策として、全ての状態操作を conditional に行い、

$$\begin{aligned} &\frac{1}{\sqrt{2}} \left(\left|0 \cdots 0\right\rangle + \frac{1}{\sqrt{\sum_k \sigma_k^{-2}}} \sum_{k=1}^{r} \frac{1}{\sigma_k} \left|\boldsymbol{v}_k\right\rangle \left|\boldsymbol{u}_k\right\rangle \right) \\ &\frac{1}{\sqrt{2}} \left(\left|0 \cdots 0\right\rangle + \left|\tilde{\boldsymbol{x}}\right\rangle \left|\boldsymbol{y}\right\rangle \right) \end{aligned} \tag{50}$$

のような状態を用意する。この二つの状態間の swap test を行えば、得られる期待値は $|\frac{1}{2} + \frac{1}{2} f_{w^*}(\tilde{\boldsymbol{x}})|^2$ となるので、$f_{w^*}(\tilde{\boldsymbol{x}})$ の符号まで復元することができる。

6.2 量子サポートベクターマシン

6.2.1 サポートベクターマシン

サポートベクターマシン自体の詳細な説明に立ち入ると 4 ページ以上は必要になると思うので他の文献に譲る。[*5]サポートベクターマシンは、与えられたデータセットの二値分類を行う手法の一つである。二値分類問題なので、各データ $\boldsymbol{x}_i$ について教師データは $y_i = \pm 1$ のどちらかの値をとることにする。サポートベクターマシンは、与えられた訓練データをもとに、$\boldsymbol{x}_i$ を分割するのに最適な超平面 $\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$ を探すアルゴリズムである。最適化された超平面の片側には $y_i = 1$ のデータのみが存在し、もう片方には $y_i = -1$ のデータのみが存在するような超平面を実現するパラメータ $\boldsymbol{w}, b$ を探索する。サポートベクターマシンが構築するモデルは

$$f_{\boldsymbol{w},b}(\boldsymbol{x}) = \mathrm{sign}\left(\boldsymbol{w} \cdot \boldsymbol{x} - b\right) \tag{51}$$

[*5] 例えば [2] の 7 章や [10] の (20) 式。ネット検索でもたくさん出てくる。

である。

サポートベクターマシンは、この問題を線型連立方程式に帰着させて解く。その際、$\boldsymbol{w}$ は以下の関係式によって変数 $\boldsymbol{\alpha}$ へと変換される。

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i \boldsymbol{x}_i \tag{52}$$

また、カーネル行列と呼ばれる行列 K を以下で定義する。

$$K_{ij} = \sum_{k=1}^{M} x_{ik} x_{jk} \tag{53}$$

これらを用いて、サポートベクターマシンが解く線型連立方程式は

$$\begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & & & & \\ 1 & & & & \\ \vdots & & & K + \gamma^{-1} I & \\ 1 & & & & \end{pmatrix} \begin{pmatrix} b \\ \boldsymbol{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ \boldsymbol{y} \end{pmatrix} \tag{54}$$

である。ここで

$$F := \begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & & & & \\ 1 & & & & \\ \vdots & & & K + \gamma^{-1} I & \\ 1 & & & & \end{pmatrix} \tag{55}$$

$$\boldsymbol{\alpha}' := \begin{pmatrix} b \\ \boldsymbol{\alpha} \end{pmatrix} \tag{56}$$

$$\boldsymbol{y}' := \begin{pmatrix} 0 \\ \boldsymbol{y} \end{pmatrix} \tag{57}$$

と定義しておく。γ は正則化パラメータと呼ばれる実数のパラメータであり、上の行列が逆行列を持つようにする役割を持つ。例えば訓練データセットが線型分離不可能であるとき、$\gamma \to \infty$ のとして正則化項 $\gamma^{-1} I$ を消してしまうと、上式は解を持たなくなる。そのような事態をさけるため、訓練データに対してある程度の分類ミスを許す効果を持つのが γ である。

ちなみに変数変換によって、モデルは

$$f_{\boldsymbol{\alpha}'}(\boldsymbol{x}) = \mathrm{sign}\left(\sum_{i=1}^{N} \alpha_i \boldsymbol{x}_i \cdot \boldsymbol{x} - b \right) \tag{58}$$

と書けることになる。

6.2.2 量子アルゴリズム

ここでは [8] にて提案された量子アルゴリズムを紹介する。大まかなアルゴリズムは以下のようになる。

1. QRAM を用いて $|\boldsymbol{y}'\rangle$ を準備する。
2. カーネル行列 K を QRAM を使って密度行列として準備する。この密度行列を $\rho_K = \sum_{ij} K_{ij} |i\rangle \langle j|$ とする。
3. Density matrix exponentiation を使って F の逆行列演算を $|\boldsymbol{y}'\rangle$ に対して行う。→最適なモデルパラメータ $\boldsymbol{\alpha}'^*$ を保持した量子状態 $|\boldsymbol{\alpha}'^*\rangle$ が得られる。
4. 未知のデータ $\tilde{\boldsymbol{x}}$ について、swap test を用いて分類を行う。

細かな点の説明を書いていく。

1 は 4.2.1 を参考にすれば良い。2 については、6.1.2 と全く同じ操作をすれば良い。$X^T X = K$ であることに注意したい。問題になるのは 3 の部分だと思う。3 は F の exponentiation を行わなければならないが、準備したのは ρ_K だけであって、F そのものではない。そこで、このアルゴリズムでは

$$J := \begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{59}$$

とし、

$$e^{F\delta} \approx e^{J\delta} e^{K\delta} e^{\gamma^{-1} I\delta} \tag{60}$$

と Trotter 展開によって $e^{2\pi F}$ を実行する。$e^{K\delta}$ は ρ_K を使って density matrix exponentiation を行えばよいし、$e^{\gamma^{-1} I\delta}$ は単純な位相ゲートとして実行可能である。$e^{J\delta}$ については、行列 J が解析的に対角化できることを用いる。J は $N-1$ 重に縮退した固有値 0 と、非ゼロの 2 つの固有値 $\lambda_\pm = \pm\sqrt{N}$ を持ち、非ゼロの固有値に対応する固有ベクトルは $\frac{1}{\sqrt{2}} \left(|0\rangle + \frac{1}{\sqrt{N}} \sum_{i=1}^N |i\rangle \right)$ である。よってこの 2 つのベクトルを計算基底に持ってくるゲートをかけたあとに、位相ゲートを作用させれば $e^{J\delta}$ を実行できる。

最後の swap test に関しては、前章と同様に符号を復元するために以下のアプローチを取る。まず、式 (58) の $\sum_{i=1}^N \alpha_i \boldsymbol{x}_i$ に相当する状態を作り出す。それをするには、ステップ 3 で作り出した状態

$$|\boldsymbol{\alpha}'^*\rangle = b |0\rangle + \sum_{i=1}^N \alpha_i |i\rangle \tag{61}$$

の $\sum_{i=1}^N \alpha_i |i\rangle$ の部分に対して、$\boldsymbol{x}_i$ をエンコードする以下のような変換を適用すればよい。[*6]

$$|i\rangle |0\rangle \xrightarrow{\text{something+QRAM}} |i\rangle |\boldsymbol{x}_i\rangle \tag{62}$$

すると、以下の状態を生成できる。

$$|\psi_{\text{model}}\rangle = b |0\rangle |0\rangle + \sum_{i=1}^N \alpha_i |i\rangle |\boldsymbol{x}_i\rangle \tag{63}$$

[*6] この変換は例えば 4.2.1 の方法を使えば実現できる。

この状態と同時に未知データを以下のようにエンコードした状態

$$|\psi_{\mathrm{data}}\rangle = \frac{1}{\sqrt{N+1}}\left(|0\rangle|0\rangle + \sum_{i=1}^{N}|i\rangle|\tilde{\boldsymbol{x}}\rangle\right) \tag{64}$$

を準備する。ただし、この 2 つの状態に対して単純に swap test を適用するだけでは、線型回帰と同じく符号が復元されない問題に陥ってしまう。そこで、[8] では、もう一つの補助ビットを用いて、conditional に $|\psi_{\mathrm{data}}\rangle, |\psi_{\mathrm{model}}\rangle$ を準備して、以下の状態を用意することが提案された。

$$\frac{1}{\sqrt{2}}\left(|0\rangle|\psi_{\mathrm{model}}\rangle + |1\rangle|\psi_{\mathrm{data}}\rangle\right) \tag{65}$$

この状態を補助ビットの X 基底で測定すれば、$f_{\boldsymbol{\alpha}'^*}$ を復元することができる。

6.3 行列の low-rank 近似サンプリング

このセクションの締めくくりに、量子推薦アルゴリズム [5] のサブルーチンとして提案された、行列の low-rank 近似を行うアルゴリズムについて紹介する。

6.3.1 特異値分解・次元削減

機械学習では、学習の前処理として、データ行列 X に対して特異値分解を行い、特異値の大きい成分のみを取り出すことがよく行われる。特異値が小さい成分は、教師データの説明に寄与しない部分であると考え切り捨てるのだ。機械学習の文脈では、この作業は**主成分分析**とか**次元削減**とも呼ばれる。実際、訓練データとして与えられるデータは少数の特異ベクトルを使えば十分再現できることが多い。

この主成分分析を定式化すると以下の様になる。

1. X の特異値分解を行う。$X = U\mathrm{diag}\left(\sigma_1 \cdots \sigma_M\right)V^\dagger$.
2. ある閾値 σ_{th} を超えた成分のみを取り出す。つまりしきい値を超えた特異値の数を k_{th} として、$X_{\mathrm{th}} := U_{\mathrm{th}}\mathrm{diag}\left(\sigma_1 \cdots \sigma_{k_{\mathrm{th}}}\right)V_{\mathrm{th}}^\dagger$ を作る。ただし $U_{\mathrm{th}}, V_{\mathrm{th}}^\dagger$ は k_{th} 本の列・行ベクトルを U, $V^\dagger$ から取ってきた行列である。
3. 次元の削減されたデータ行列 X_{th} を使って、好きな機械学習アルゴリズムを走らせる。

量子コンピュータによってこの作業を加速することが可能である。

6.3.2 量子アルゴリズム

4.2.2 で紹介した Prakash 法を用いて量子ウォーク (5.4 参照) を構築することで実現する。大まかなアルゴリズムを先に示そう。

1. X を 5.4 における D として持つ量子ウォーク W を作る。
2. $\frac{1}{\sqrt{N}}\sum_{i=1}^{N}|\boldsymbol{x}_i\rangle|i\rangle$ に対して W の位相推定を作用させる。$\sum_{k=1}^{r}|\mu_{k,\pm}\rangle\left|b(\sigma_k^2)\right\rangle$ が得られる。

3. 補助ビットを追加し、σ_k^2 が σ_{th} よりも大きいところ場合のみ $|0\rangle$、それ以外の場合のみ $|1\rangle$ とするような演算を行う。$\sum_{k=1}^{k_{\mathrm{th}}} |\mu_{k,\pm}\rangle \left|b(\sigma_k^2)\right\rangle |0\rangle + \sum_{k=k_{\mathrm{th}}+1}^{r} |\mu_{k,\pm}\rangle \left|b(\sigma_k^2)\right\rangle |1\rangle$ を得る。
4. 補助ビットを測定する。$|0\rangle$ が出れば、$\sum_{k=1}^{k_{\mathrm{th}}} |\mu_{k,\pm}\rangle \left|b(\sigma_k^2)\right\rangle |0\rangle$ となり、σ_{th} 以上の成分のみを取り出せる。
5. ステップ 1・2 を uncompute する。すると $\frac{1}{\sqrt{N}} \sum_{i=1}^{N} |\boldsymbol{x}_{\mathrm{th},i}\rangle |i\rangle$ が得られる。

ここで $|\mu_{k,\pm}\rangle$ は 5.4 で定義した量子ウォーク W の固有ベクトルで、$\boldsymbol{x}_{\mathrm{th},i}$ は X_{th} の行ベクトルである。

ほとんどの部分は 6.1.2 で説明した手法と同様の手法になるので説明は省略するが、ステップ 1 の量子ウォーク W を作る部分だけは独立していると思うので説明する。W を作るには、以下のユニタリー操作 P と Q を使えば良い。

$$P |0\rangle |i\rangle = |\boldsymbol{x}_i\rangle |i\rangle \tag{66}$$

$$Q |j\rangle |0\rangle = |j\rangle \frac{1}{\sqrt{N}} \sum_{i=1}^{N} |i\rangle \tag{67}$$

P は Prakash 法 (4.2.2 参照) によって構成する。P, Q によって生成できる状態 $|\boldsymbol{x}_i\rangle |i\rangle$ と $|j\rangle \frac{1}{\sqrt{N}} \sum_{i=1}^{N} |i\rangle$ の内積を取ってみると、

$$(\langle \boldsymbol{x}_i| \langle i|) \left(|j\rangle \frac{1}{\sqrt{N}} \sum_{k=1}^{N} |k\rangle \right) = \frac{x_{ij}}{\sqrt{N}} \tag{68}$$

となる。5.4 と見比べると、この 2 つのベクトルを $|\boldsymbol{v}_i\rangle$, $|\boldsymbol{w}_j\rangle$ として使って W を構成すればよいことがわかるだろう。式 (36) のようなゲートを構築するには、以下の事実を用いる。

$$\sum_{i=1}^{N} |\boldsymbol{x}_i\rangle |i\rangle \langle \boldsymbol{x}_i| \langle i| = \sum_{i=1}^{N} P |0\rangle |i\rangle \langle 0| \langle i| P^\dagger \tag{69}$$

$$= P |0\rangle \langle 0| \otimes \left(\sum_{i=1}^{N} |i\rangle \langle i| \right) P^\dagger \tag{70}$$

$$= P |0\rangle \langle 0| \otimes I_N P^\dagger \tag{71}$$

(I_N は N 次元の単位行列。) これを使うと、

$$2 \sum_{i=1}^{N} |\boldsymbol{x}_i\rangle |i\rangle \langle \boldsymbol{x}_i| \langle i| - I_{MN} = P \left(2 |0\rangle \langle 0| - I_M\right) \otimes I_N P^\dagger \tag{72}$$

が得られる。よって $2 \sum_{i=1}^{N} |\boldsymbol{x}_i\rangle |i\rangle \langle \boldsymbol{x}_i| \langle i| - I_{MN}$ は $P, P^\dagger$ と conditional 位相ゲートによって構成できる。もう片方についても同様の手法で構成可能である。

7 NISQ デバイスによる機械学習

7.1 Noisy intermediate scale quantum デバイス

Noisy intermediate scale quantum、略して NISQ デバイスとは、スーパーコンピュータをもってしてもシミュレーションが難しくなるような qubit 数・ゲート精度を兼ね備えているが、無視できないレベルのノイズの影響下にある量子デバイスのこと。誤り訂正を備えていないため、上で紹介したようなアルゴリズムを実行することは不可能だし、QRAM に至っては夢のまた夢である。しかしある意味でスーパーコンピュータを超えた計算能力を持っていることは確かなわけなので、これを何らかの形で応用しようという試みが広く研究されている。ここでは機械学習という文脈で最近の研究を紹介する。

7.2 変分量子アルゴリズム

NISQ デバイスを応用するために考えられた手法は、ほとんどが**変分量子アルゴリズム**という枠組みに入る。変分量子アルゴリズムと呼ばれているものの大まかなアルゴリズムは以下のようになる。

1. 適当なパラメータ $\boldsymbol{\theta}$ を持つユニタリーゲート $U(\boldsymbol{\theta})$ を初期状態 $|0\rangle$ に作用させ、$|\psi(\boldsymbol{\theta}\rangle := U(\boldsymbol{\theta})\,|0\rangle$ を作る。
2. $|\psi(\boldsymbol{\theta}\rangle$ について何らかのオブザーバブルを測定する。
3. 測定結果によってパラメータを更新し、$\boldsymbol{\theta}'$ を使って 1, 2 を繰り返す。

$U(\boldsymbol{\theta})$ は NISQ デバイスでも実行できるような、比較的簡単なものでなければならない。そのような簡単なものであっても、qubit 数が十分多くなれば $|\psi(\boldsymbol{\theta})\rangle$ は古典コンピュータで表現できないと信じられている。変分量子アルゴリズムには、このことを存分に使うようなプロトコルが望まれている。

7.3 Quantum variational autoencoder

Variational autoencoder (VAE) はディープニューラルネットの事前チューニングの手法として提案された、次元削減のためのアルゴリズムである。その量子版として、quantum variational autoencoder (QVAE) がある。QVAE は、n qubit の状態として与えられた複数の状態 $\{|\varphi_i\rangle\}_{i=1}^N$ を $m < n$ qubit の状態へと圧縮するためアルゴリズムである。そのアルゴリズムを以下に示す。

1. $|\varphi_i\rangle$ それぞれに対して $U(\boldsymbol{\theta})$ を作用させる。$|\psi_i\rangle := U(\boldsymbol{\theta})\,|\varphi_i\rangle$ とする。
2. $|\psi_i\rangle$ のうち、$m-n$ qubit を Z 基底で測定する。測定後の状態を $|\psi_i'\rangle$ とする。
3. $|\psi_i'\rangle$ それぞれに対して $U^\dagger(\boldsymbol{\theta})$ を作用させる。
4. $|\langle\varphi_i|U^\dagger(\boldsymbol{\theta})|\psi_i'\rangle|^2$ を測定する。

5. $\sum_{i=1}^{N} |\langle\varphi_i|U^\dagger(\boldsymbol{\theta})|\psi_i'\rangle|^2$ が大きくなるようにパラメータを更新し繰り返す。

もし $U(\boldsymbol{\theta})$ が $|\varphi_i\rangle$ の全てに対して、その情報を m qubit の部分系に圧縮することができていれば、$U^\dagger(\boldsymbol{\theta})$ によって量子状態が完全に復元されるはずなので、理想的には $|\langle\varphi_i|U^\dagger(\boldsymbol{\theta})|\psi_i'\rangle|^2 = 1$ となるはずである。QVAE の限界は、$\rho = \frac{1}{N}\sum_i |\varphi_i\rangle\langle\varphi_i|$ とするとき、ρ の rank によって決まる。

7.4 量子回路学習

量子回路学習 (quantum circuit learning, QCL) は NISQ 上で教師ありの機械学習を行うためのアルゴリズムである。訓練データは説明変数 $\{\boldsymbol{x}_i\}_{i=1}^N$ と対応する教師データ $\{y_i\}_{i=1}^N$ のセットとする。そのアルゴリズムを以下に示す。

1. それぞれの $\boldsymbol{x}_i$ について、適当な量子ゲート $V(\boldsymbol{x}_i)$ を使って、データをエンコードした状態 $|\varphi(\boldsymbol{x}_i)\rangle := V(\boldsymbol{x}_i)|0\rangle$ を作る。
2. パラメータ付きの量子回路 $U(\boldsymbol{\theta})$ を使い、$|\psi(\boldsymbol{x}_i,\boldsymbol{\theta})\rangle := U(\boldsymbol{\theta})|\varphi(\boldsymbol{x}_i)\rangle$ を作る。
3. $|\psi(\boldsymbol{x}_i,\boldsymbol{\theta})\rangle$ に対して、適当なオブザーバブルを観測する。例えば 1 qubit 目の Z 期待値。
4. 上記で観測した値と教師データ y_i の差が小さくなるようにパラメータ $\boldsymbol{\theta}$ を調節し、1 に戻る。

このアルゴリズムが NISQ をどのように使っているのか説明して、このノートの締めとする。このアルゴリズムが構築するモデルは、例えば 1 qubit 目の Z 期待値を出力として測定するとき、

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \langle\psi(\boldsymbol{x},\boldsymbol{\theta})|Z_1|\psi(\boldsymbol{x},\boldsymbol{\theta})\rangle \tag{73}$$

である。このモデル $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ はどのような構造を持つか調べよう。

簡単な例として、$\boldsymbol{x}$ がスカラー $\boldsymbol{x}$ であるときを考えよう。入力ゲート $V(x)$ としては、量子回路をできるだけ浅くするため、1 qubit の回転ゲート

$$V(x)|0\rangle = x|0\rangle + \sqrt{1-x^2}|1\rangle \tag{74}$$

を独立に n qubit それぞれに作用させることを考えよう。このとき $|\varphi(x)\rangle$ は

$$|\varphi(x)\rangle = \left(x|0\rangle + \sqrt{1-x^2}|1\rangle\right)^{\otimes n} \tag{75}$$

となる。テンソル積構造によって、元のデータ x に対する非線形性が生まれていることに注目したい。このように生まれた非線形項が、$U(\boldsymbol{\theta})$ によって線型に組み合わされてモデル $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ に現れる。量子状態には一般に 2^n 個の係数を埋め込むことができるため、NISQ を使うことによって、指数関数的に多数の基底関数によって、$f_{\boldsymbol{\theta}}(\boldsymbol{x})$ を構成できる。これが QCL の利点となる可能性のあるポイントである。

参考文献

[1] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, Vol. 549, No. 7671, pp. 195–202, sep 2017.

[2] Christopher M Bishop. *Pattern Recognition and Machine Learning*. New York, aug 2006.

[3] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Architectures for a quantum random access memory. *Phys. Rev. A*, Vol. 78, No. 5, p. 052310, nov 2008.

[4] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions. *arXiv preprint*, 2002.

[5] Iordanis Kerenidis and Anupam Prakash. Quantum Recommendation Systems. *arXiv preprint*, 2016.

[6] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nat. Phys.*, Vol. 10, No. 9, pp. 631–633, sep 2014.

[7] Anupam Prakash. *Quantum Algorithms for Linear Algebra and Machine Learning*. PhD thesis, EECS Department, University of California, Berkeley, Dec 2014.

[8] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum Support Vector Machine for Big Data Classification. *Phys. Rev. Lett.*, Vol. 113, No. 13, p. 130503, sep 2014.

[9] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. Prediction by linear regression on a quantum computer. *Phys. Rev. A*, Vol. 94, No. 2, p. 022342, aug 2016.

[10] J.A.K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, Vol. 9, No. 3, pp. 293–300, Jun 1999.