

A walk through of time series analysis on quantum computers

Ammar Daskin

Abstract—Because of the rotational components on quantum circuits, some quantum neural networks based on variational circuits can be considered equivalent to the classical Fourier networks. In addition, they can be used to predict the Fourier coefficients of continuous functions. Time series data indicates a state of a variable in time. Since some time series data can be also considered as continuous functions, we can expect quantum machine learning models to do many data analysis tasks successfully on time series data. Therefore, it is important to investigate new quantum logics for temporal data processing and analyze intrinsic relationships of data on quantum computers.

In this paper, we go through the quantum analogues of classical data preprocessing and forecasting with ARIMA models by using simple quantum operators requiring a few number of quantum gates. Then we discuss future directions and some of the tools/algorithms that can be used for temporal data analysis on quantum computers.

Index Terms—time series, temporal data, quantum machine learning, quantum algorithms, quantum optimization

I. INTRODUCTION

It is shown that nature itself is explained better in quantum physics than classical one. The idea of quantum computers was put forward by Feynman [1] and other pioneers [2] so as to simulate and understand intrinsic physical properties of nature in a better way: Simulating quantum systems with many particles is difficult on classical computers because the computational complexity in general grows with the number of involved particles. On the other hand, quantum computers are based on qubits implemented by quantum particles which in turn indicates that in principle, we can increase the computational power of quantum computers exponentially by adding controllable new qubits. Therefore, the difficulty of simulating nature for classical computers would become the main advantage of quantum computers. Unfortunately since the input/output of quantum computers are still in classical ones and zeros, it necessitates a state preparation that involves classical processing of the data and a measurement that requires many repetitions of the same computation. Therefore, the complexity boundaries in classical computation mostly remain the same for quantum computing. Nonetheless, it is still possible to design more efficient algorithms by employing quantum entanglement and superposition (quantum parallelism): Some known examples are Shor's integer factoring algorithm [3], Grover's search algorithm in an unordered list [4], and the quantum principal component analysis (PCA) [5].

A. Classical neural networks

The representation theorem indicates any multivariate continuous function can be represented as a superposition of

functions of one variable and addition [6, 7]. An artificial neural network can be considered as a composition of simple nonlinear activation function[8]: Let I^n be the n-dimensional unit-cube $[0, 1]^n$ and $C(I^n)$ be the space of continuous functions on I^n . In addition, let σ be a sigmoidal function. Given any $f \in C(I^n)$ and $\epsilon > 0$, there is a G of the following form:

$$G(x) = \sum_{j=1}^n \alpha_j \sigma(w_j^T x + \theta_j) \quad (1)$$

for which $|G(x) - f(x)| < \epsilon$.

Gallant et al. [9] showed a single feed-forward neural network where the output is squashed by using a cosine activation function: i.e. by considering σ as cosine, sine, or exponential (i.e. in the form $e^{iw_j^T x + \theta_j}$) functions, one can approximate the Fourier transform of a continuous function. These networks are often called Fourier neural networks (FNN). On the other hand, Fourier analysis of data also sheds lights on the generalization performances of deep neural networks[10].

B. Time series problems

Temporal data indicates a state of a variable in time, which could be collected through a wide variety of monitoring devices. Examples include financial market data; sensor data which may indicate temperature, humidity and so on; and medical data collected by different monitoring medical devices. Depending on the collection method, the data can be continuous or discrete. Continuous temporal data sets are generally referred to as times series data. [11] The data mining tasks on time series data include clustering, classification, and forecasting: In general it is required to specialize the known algorithms in data analysis in order to use in the analysis of time series data: e.g. see the classification with deep learning for time series data [12] and the distance based time series classification [13]. It is shown that time series data can be also modeled by using Fourier deep neural networks [14], which is briefly explained above.

C. Quantum machine learning

Since the quantum machine learning [15, 16] is a very new field, there are only a small number of partial studies which can be found for the analysis of time series data on quantum computers: the time series data analysis is done by using a quantum approach based on the Schrödinger's equation [17]. In a very recent paper, the quantum machine learning approaches are investigated for the problems in finance [18]. It is shown that the variational quantum machine learning

models can be used to approximate Fourier transforms of the continuous functions [19]. It is also shown that the continuous variable (CV) quantum neural networks can be used for curve fitting and function approximations[20]. It is still important to study and build generic separate models and packages for temporal data analysis.

Quantum circuits are in general based on quantum gates described by some form of rotations in the Hilbert space. For instance, the rotation gates for one-qubit can be defined in the following forms;

$$R_x(2\theta) = \begin{pmatrix} \cos(\theta) & -i\sin(\theta) \\ -i\sin(\theta) & \cos(\theta) \end{pmatrix}, \quad (2)$$

$$R_y(2\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}, \quad (3)$$

$$\text{and } R_z(2\theta) = \begin{pmatrix} e^{-i\theta} & 0 \\ 0 & e^{i\theta} \end{pmatrix}. \quad (4)$$

Because of these rotational components, the quantum circuits can in theory provide a natural equivalent to the classical Fourier networks (e.g., see our previous paper[21] which gives an example quantum neural net with periodic activation function.). In addition, they can be used to predict Fourier coefficients of continuous functions as done in Ref.[20]

D. Quantum temporal data analysis

Since some time series data can also be considered as continuous functions, quantum machine learning models can successfully do many data analysis tasks on the time series data. Therefore, it is important to investigate new quantum logics for temporal data processing and analyze intrinsic relationships of data on quantum computers. The processing data on quantum computers in general requires the followings:

- classical preprocessing and quantum state preparation,
- quantum preprocessing,
- quantum processing of data,
- quantum measurement,
- and analyses of the measurement results on classical computers

In general, all or some of these steps are repeated until a desired result/model is obtained in data analysis. For continuous temporal data, these steps are visualized in Fig. 1.

In the following sections, we go through the quantum analogues of classical data preprocessing and forecasting. Then we discuss some of the tools/algorithms that can be used for temporal data analyses on quantum computers and discuss future directions.

II. PREPARING DATA FOR QUANTUM ALGORITHMS

The time series data is generally given in contextual representations where attributes describe the context and behavioral values at any reference points [11]. If more than one behavioral attribute is associated with the series, then the data is referred as the multivariate time series: i.e. A time series data with dimension d and length n . The data is formed by a set of the

values received at the time stamp t_i . In the i th time stamp, the received data is a vector of dimension d : $\mathbf{y}_i = (y_i^1, \dots, y_i^d)$.

In classical data analyses, the time series data is taken under preprocessing operations: e.g., the missing values may be handled, the data may be normalized, smoothed, or scaled. Then, a variety of methods are used to transform and reduce the dimension of the data. After these steps, the standard data analysis tasks such as classification and clustering are also applicable to time series data. The classification and clustering tasks involve determining the correlation across the series and classifying the series based on their shapes. These can be offline or real time. And class labels may need to be assigned to the individual time stamps (point labels) or the whole series.

For quantum computers, the algorithms either use the data after some classical preprocessing, or as done in PCA [5], it assumes the data is processed and stored in quantum RAM [22]. In the former case, the data is prepared as a quantum state by a state preparation circuit that converts a standard initial state: e.g. $|0\rangle$, into the state that represents the data $|x\rangle$. If the length of $|x\rangle$ is n , then this kind of state preparation requires a circuit with $O(n)$ number of controlled-NOT quantum gates in general [23]. In some cases, the data is also used as parameters that determine the rotation angle of the quantum gates (e.g. in [24], the data is used with parameters to define quantum rotation gates. The learning task is done by reusing the data and adjusting the value of the parameters for quantum gates.). After the mapping, $|x\rangle$ is a vector defined in the Hilbert space. It is shown in Ref.[25] that the Hilbert space can be considered as a kernel space used in classical machine learning to ease the training process. Therefore, the training of a learning model can be achieved also in the Hilbert space. Although this solves the representability of the data in the Hilbert space, it is still necessary to design efficient data mapping methods without disrupting the representability of the data (e.g. for MNIST data, it is reduced to 8 qubits by using classical PCA [26] which reduces the training results to as low as 70%).

Consequently, the representation is highly dependent on the quantum algorithm to be used in the data processing step. However, in most cases, since the time series data is multivariate in the form $\mathbf{y}_i = (y_i^1, \dots, y_i^d)$, the quantum superposition approach can be used to efficiently represent the data on quantum computers. This in general may reduce the $d \times n$ data into an n -dimensional quantum state:

$$|y\rangle = \frac{1}{\sqrt{n}} \sum_i^n |y_i\rangle \quad (5)$$

Alternatively, one can build the following quantum state from the input data:

$$|y\rangle = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad (6)$$

Here, the dimension of the quantum state is $\log(nd)$. Another alternative can be the following:

$$|y\rangle = |y_1\rangle |y_2\rangle \dots |y_n\rangle. \quad (7)$$

d-dimensional classical Multivariate data
observed in time

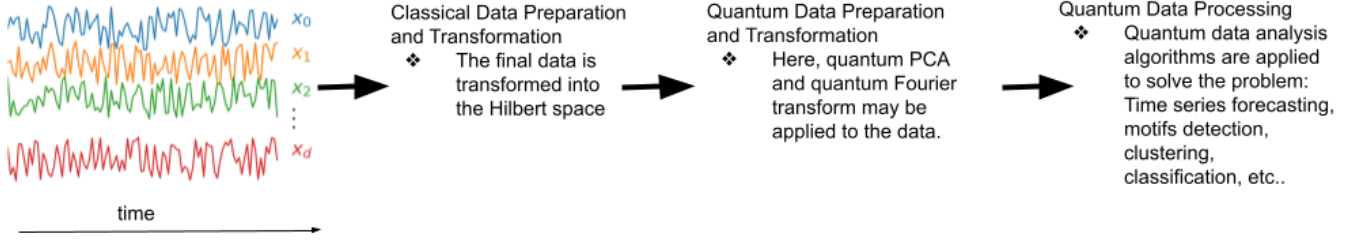


Fig. 1. General picture for processing temporal data on quantum computers (the measurement step is not shown.)

Although the required number of qubit which is $n \log(d)$ is much more than the first two methods, this representation may allow us to apply quantum operations to individual set of values more easily. However, since the required number of qubits depend on n , it may not be possible on current quantum computers to use this representation for the series with large n .

III. QUANTUM PREPROCESSING OF TEMPORAL DATA

Below we show how to implement quantum analogues of some of the famous classical preprocessing techniques: the exponential smoothing, binning and data transformation. Note that in most cases, we go through the steps for univariate data.

A. Exponential smoothing

In the exponential smoothing, the smoothed value y'_i is obtained through the combination of y_i and the pervious values with the ratios determined by the decay factor α :

$$y'_i = \alpha y_i + (1 - \alpha) y'_{i-1}. \quad (8)$$

The recursive substitution in the above equation yields:

$$y'_i = (1 - \alpha)^i y'_0 + \alpha \sum_{j=1}^i y_j (1 - \alpha)^{i-j}. \quad (9)$$

In the analyses, the choice of α determines the impact of the previous data points.

In quantum analyses, we can construct a similar exponential smoothing technique:

- 1) Different from the classical analysis, $|y_i\rangle$, the data observed at the i th time stamp, can be a single data point or multiple data points observed on d dimension. Therefore, as a physical qubit, this state can be implemented as a d -dimensional quantum system with $O(\log(d))$ qubits or a d -level qubit.
- 2) The new observed data at time stamp $(i+1)$ is prepared as a quantum state $|y_{i+1}\rangle$ on different qubits: i.e., $|y_{i+1}\rangle|y_i\rangle$. Or we can create the following quantum state by using a control qubit and data qubit.

$$\begin{pmatrix} |y_{i+1}\rangle \\ |y_i\rangle \end{pmatrix} \quad (10)$$

Note that this state can be formed with the help of control qubit by applying the operators related to $|y_{i+1}\rangle$ and $|y_i\rangle$ to the data qubit in $|0\rangle$ state.

- 3) From the above state, a quantum entangling gate (e.g. the Hadamard gate) can be used to generate the following recurrence:

$$\begin{aligned} |y'_{i+1}\rangle &= \sqrt{\alpha} (|y_{i+1}\rangle + |y_i'\rangle) + \sqrt{1-\alpha} (|y_{i+1}\rangle - |y_i'\rangle) \\ &= a |y_{i+1}\rangle + b |y_i'\rangle \end{aligned} \quad (11)$$

where $a = (\sqrt{\alpha} + \sqrt{1-\alpha})$ and $b = (\sqrt{\alpha} - \sqrt{1-\alpha})$. In the case of the Hadamard gates, α is some power of $1/2$. Using the above recursion, if we substitute $|y_i'\rangle$ into the above, $|y'_{i+1}\rangle$ becomes:

$$\begin{aligned} |y'_{i+1}\rangle &= a |y_{i+1}\rangle + b (a |y_i\rangle + b |y'_{i-1}\rangle) \\ &= a |y_{i+1}\rangle + ab |y_i\rangle + ab^2 |y'_{i-1}\rangle \dots \\ &= ab^{i+1} |y'_0\rangle + \sum_{k=1}^i ab^{i-k} |y_{k+1}\rangle, \end{aligned} \quad (12)$$

If we choose an α which makes $b < 1$, we can have the quantum equivalent of the classical exponential smoothing. In this case, the choice of α regulates the decay of the terms through b ; therefore, we can call b as the decay factor that determines the importance of the recent data points in any analysis using $|y'_i\rangle$.

B. Binning and moving average smoothing

The classical moving average is a special case of the binning method where the date is divided into equally spaced time intervals of size k : $[t_1, t_k], [t_{k+1}, t_{2k}]$. Then, the data points are assigned into the corresponding bins and the mean values of the bins are used for the analyses:

$$y'_{i+1} = \frac{\sum_{r=1}^k y_{i,k+r}}{k} \quad (13)$$

Here, the number of the data points are reduced by a factor of k .

Utilizing the Hadamard gates on certain qubits, one can obtain the average of the neighboring states on certain quantum

states. After finding the average, we can also disregard the rest of the state by collapsing quantum state onto the interested part of the system: One example could be as follows:

- Let $|y_i\rangle = \begin{pmatrix} y_{i1} \\ \vdots \\ y_{id} \end{pmatrix}$. If we multiply this quantum state $U_{bin} = I^{\otimes \log_2(d/k)} \otimes H^{\otimes \log_2 k}$, where H is the Hadamard gate:

$$H = 1/\sqrt{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (14)$$

If we apply U_{bin} to the quantum state $|y_i\rangle$, then when the first $\log_2 k$ qubits in $|0 \dots 0\rangle$ state, we would have an equivalent y'_{i+1} which could be considered reduced data points.

The classical moving average smoothing uses overlapping bins which can be easily done on quantum case by utilizing a quantum operator of the form: $H^{\otimes \log_2 k} \otimes I^{\otimes \log_2(d/k)}$.

C. Data transformation

In classical data analyses, often the temporal data is made smaller or discretized through data transformation methods such as discrete wavelet and Fourier transforms (DWT and DFT respectively).

Similar to the Fourier transform, DWT represents a data (function) in terms of an orthonormal basis. One of the simplest wavelet transform is the Haar transform which can be described a orthogonal matrix: e.g., 4×4 Haar transform is

$$H_4 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix} \quad (15)$$

Similarly to Walsh-Hadamard transform, the unnormalized Haar transform matrix can be obtained by the following recurrence:

$$H_{2N} = \begin{bmatrix} H_N \otimes [1, 1] \\ I_N \otimes [1, -1] \end{bmatrix}. \quad (16)$$

The transformation on an input vector y' is described by $y' = H_N x$. From this product, the rows of the Haar matrix allow us to draw different properties of the temporal data: For instance, for H_4 , the first row takes the average of the input vector, while the second row measures the low frequencies. The rest of the rows measures different parts of the input vector.

In the wavelet transform, y can be represented with the basis vectors that form H_n :

$$y' = \sum_{i=1}^q a_i w_i, \quad (17)$$

where w_i is the i th basis vector and a_i is the normalized coefficient. The dimension reduction is done by disregarding the lowest a_i s.

In quantum computing, since the Haar matrix is orthogonal and can be built from the recurrence that is described by the

Kronecker product, it can be implemented as a quantum gate in a similar fashion to the Hadamard gate. Therefore, given $|y\rangle$, one can easily obtain $|y'\rangle$. The elimination of lowest parts of $|y'\rangle$ can be done through either sampling or measuring a few qubits. This leads to a problem of finding the best qubits that gives the maximum reduction with the minimum information loss.

1) *Quantum Fourier transform (QFT)*: The discrete Fourier transform (DFT) can be used to transform a data vector into a linear combination of sinusoidal series. The similarity of two vectors can be measured by taking the Euclidean distance of the Fourier coefficients. In quantum computing, DFT-which has the complexity bound $O(n \log n)$ for n -dimensional vector-can be implemented with $O(\log n)$ computational steps. It provides the main speedup of many quantum algorithms such as integer factoring. The quantum Fourier transform for the data $|y\rangle$ can be obtained by applying the operation QFT . If we have two data vectors y_1 and y_2 of the same dimension n (we assume this for convenience and brevity in notations, it can be easily generalized to any dimension.), we first construct the following quantum state:

$$|y\rangle = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \quad (18)$$

Then we apply the quantum Fourier transform to each part of the vectors:

$$\begin{pmatrix} QFT & \\ & QFT \end{pmatrix} |y\rangle = \begin{pmatrix} QFT y_1 \\ QFT y_2 \end{pmatrix} \quad (19)$$

Here note that

$$\begin{pmatrix} QFT & \\ & QFT \end{pmatrix} = I \otimes QFT, \quad (20)$$

where I is a 2×2 identity matrix. We then apply the Hadamard gate to the first qubit:

$$\left(H \otimes I^{\otimes \log(n)} \right) \begin{pmatrix} QFT y_1 \\ QFT y_2 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} QFT y_1 + QFT y_2 \\ QFT y_1 - QFT y_2 \end{pmatrix} \quad (21)$$

We can also start with a superposition of the input state:

$$|y\rangle = \frac{1}{\sqrt{2}} (|y_1\rangle - |y_2\rangle). \quad (22)$$

Then we apply QFT :

$$QFT |y\rangle = \frac{1}{\sqrt{2}} (QFT |y_1\rangle - QFT |y_2\rangle). \quad (23)$$

This also generates a kind of the desired distance.

As mentioned before, we can also use a separate register for each data point at different time: e.g., the quantum state $|y_1\rangle|y_2\rangle$ is for the data measured at time $t=1$ and $t=2$. In this case we apply QFT to both registers and then use the SWAP test to create the distance measure between data points.

$$(QFT \otimes QFT) |y_1\rangle |y_2\rangle = QFT |y_2\rangle QFT |y_2\rangle \quad (24)$$

In all cases, the Euclidean distance can be obtained by the measurement results. However, if d number of y_i are given as classical input, the first approach requires $O(\log(dn))$ number

of qubits and the third approach requires $O(d \log(n))$ since a separate register is used for each y_i . Since the second approach uses the superpositioned state, the number of qubits is $O(\log(n))$ which is less than the first and third approaches. On the other hand, the classical processing time is almost the same for all approaches and is $O(nd)$ since it requires linear scanning of the every data points. However, if $|y_i\rangle$ are stored on a random accessed quantum memory (qRAM), then all approaches requires only $O(d)$ number of queries to qRAM to load the data into quantum registers.

IV. TIME SERIES FORECASTING

If the statistical parameters of time series such as the mean and variance do not change with time, it is called stationary: i.e. the probabilistic distribution of the parameters in any time interval is the same as or very close to the time interval found by shifting from this interval. And non-stationary if the distributions change with time, which is mostly the case when we deal with real world data: e.g., if we look at the number of daily COVID cases, we see that the daily cases changes based on the season and different factors for the past few years. A nonstationary series can be made stationary by different operations on the times series data.

A. Differencing

Time series can be converted by considering differences between the data y_i at time t_i and the data y_{i-1} at the previous time t_{i-1} . This operation is called differencing and defined as follows:

$$y'_i = y_i - y_{i-1}. \quad (25)$$

If the data is still not stationary, then a second order differencing also can be applied:

$$y''_i = y'_i + y'_{i-1} = y_i - 2y_{i-1} + y_{i-2} \quad (26)$$

To implement this as a quantum operation, first we observe that the closer $|y'\rangle$ to the equal superposition state, $|H\rangle = H^{\otimes n}|0\rangle$, the more stationary it is. We can measure the closeness of the two quantum state by the swap test[27] or by some statistical sampling from the quantum state(i.e., measuring some qubits to obtain probabilities of 0 and 1. The same or close values indicate an equal superposition state).

To construct $|y'\rangle$ first we use an auxiliary qubit in $|1\rangle$, then apply a Hadamard gate to this qubit:

$$(H|1\rangle)|y\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} |y\rangle \\ -|y\rangle \end{pmatrix} \quad (27)$$

Then we apply laddered Toffoli operations (similar to a bit-adder: See quantum adders in Ref.[23]) to the second part of

the state to convert into the following:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} |y\rangle \\ -U_{adder}|y\rangle \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \\ -y_{n-1} \\ -y_0 \\ \vdots \\ -y_{n-2} \end{pmatrix} \quad (28)$$

Then, we apply a Hadamard gate again to the first qubit to obtain the following

$$\left(H \otimes I^{\otimes n-1} \right) \frac{1}{\sqrt{2}} \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \\ -y_{n-1} \\ -y_0 \\ \vdots \\ -y_{n-2} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} y_0 - y_{n-1} \\ y_1 - y_0 \\ \vdots \\ y_{n-1} - y_{n-2} \\ y_0 + y_{n-1} \\ y_1 + y_0 \\ \vdots \\ y_{n-1} + y_{n-2} \end{pmatrix} \quad (29)$$

Note that by applying Hadamard gates on different qubits and some swap operations we can easily built $|y''\rangle$ or any high order differencing from the above state. For instance, one can obtain some seasonal differencing as follows: For a natural number $s < i$, let the seasonal difference be defined as

$$y'_i = y_i - y_{i-s}. \quad (30)$$

In Eq.(29); instead of applying the Hadamard gate to the first qubit, if we apply to the second or any other qubit based on the value of s , we basically construct the equivalent quantum state for seasonal difference. Also note that the desired quantum state is the half part of the above state where the first qubit is in $|0\rangle$ state. In the further computations, one can either collapse the state to the desired part by measuring the first qubit in $|0\rangle$ state¹.

B. Forecasting Models

Differencing mean the time series is drifting in time; therefore, it can be modeled by simply the following:

$$y_{i+1} = y_i + c + e_{i+1}, \quad (31)$$

where y_{i+1} is the predicted value, e_{i+1} represents the white noise with zero mean, and c is the time drift.

Forecasting on time series data can be made through autoregressive models. The forecasting with the univariate time series data is generally based on autoregression models where the output value at time t , y_t , is determined as a linear combination of the values preceding window of some length p (AR(p) model)[11]:

$$y_t = \sum_{i=1}^p a_i \cdot y_{t-i} + c + \epsilon_t, \quad (32)$$

¹For positive y , we can also apply an oblivious-amplitude amplification [28] to the first qubit or consider a different order of differencing

where c and the coefficients $a_1 \dots a_p$ are learned through a learning process.

In the moving average model (MA(q)), the behavioral attribute value at any timestamp is determined from the unexpected historical variations in the time series data (shocks):

$$y_t = \sum_{i=1}^q b_i \cdot \epsilon_{t-i} + c + \epsilon_t, \quad (33)$$

where c is the mean value and the coefficients b_i s are learned from the data. A more powerful and general model can be obtained by combining the two aforementioned models (ARMA(p, q)):

$$y_t = \sum_{i=1}^p a_i \cdot y_{t-i} + c + \epsilon_t + \sum_{i=1}^q b_i \cdot \epsilon_{t-i} + c + \epsilon_t. \quad (34)$$

Autoregressive moving average models (ARMA) are used best with the stationary data (when the mean, variance, and autocorrelation does not change in time). The non-stationary data are handled by integrating a series of differences into the model (ARIMA(p, q, d)). For instance with the first-order difference ($d = 1$) the model can be written as:

$$y'_t = \sum_{i=1}^p a_i \cdot y'_{t-i} + c + \epsilon_t + \sum_{i=1}^q b_i \cdot \epsilon_{t-i} + c + \epsilon_t. \quad (35)$$

We can easily generate the above by using a variational quantum circuit including two and single qubit gates (as done in many of quantum machine learning models) and apply to the quantum state prepared in one of the following forms:

$$|y\rangle |\epsilon\rangle, (\theta_y |y\rangle + \theta_\epsilon |\epsilon\rangle), \text{ or } \begin{pmatrix} |y\rangle \\ |\epsilon\rangle \end{pmatrix}. \quad (36)$$

Note that $|y\rangle$ may also represent the state after differencing. Also note that the angle values of the gates in the circuit are related to the parameters \mathbf{a} and \mathbf{b} so that we can obtain the parameterized model in Eq.(35) or any of the previous models. Furthermore, note that following the Ref.[29] where it is shown how to map the Schmidt decomposition of a vector into a quantum circuit, one can also prepare a quantum operator whose first row is the parameters \mathbf{a} and \mathbf{b} .

$$\begin{pmatrix} a_1 & \dots & a_p & b_1 & \dots & b_q \\ \bullet & \dots & \bullet & \dots & \bullet \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \quad (37)$$

If the dimensions of \mathbf{a} and \mathbf{b} are k , this operator can be generated by using $O(k)$ number of quantum operations. We map the parameters to the angles of the rotation gates. Then, we adjust the rotation gate parameters by using a classical optimization technique as done in the variational solvers.

Since the value of p determines the seasonality (period) in the time series. Therefore, ARIMA model indicates the auto-correlation between a value and its neighbors. In some ARIMA like models, p can be considered as the period of the time series. On quantum computers the value of p can be found along with the other optimization parameters by using a classical optimization. In addition, since it is the period of the data, one can also adapt quantum phase estimation algorithm to find this period as done in Shor's factoring algorithm.

V. DISCUSSION AND FUTURE DIRECTIONS

In this paper, we only go through data processing and forecasting with ARIMA models. As also discussed at the beginning of the paper, more studies are needed to investigate algorithms and methods for preprocessing and transformation of time series data into the Hilbert space in which quantum computers work; and investigate quantum machine learning methods and algorithms for univariate and/or multivariate time series data to forecast, classify and cluster the data. In addition, for time series data, there are many classical packages such as Facebook's- Prophet [30] and ARIMA (Autoregressive Integrated Moving Average) models. There is also need to design a quantum packages-that can be used with the current quantum libraries such as IBM-QISKIT[31]-for time series data to forecast and do other tasks.

A. Data representation and Measurement

Representation of data on quantum computers may not be without noise. Therefore, the small changes in any feature of data may not be observed on quantum state by using simple measurements. If one qubit is used for one feature, then we expect any change in the input data to impact the measurement statistics of a qubit. However, representing whole feature vector as a quantum state with fewer qubits enforces an automatic dimension reduction in the measurement and hence may cause a significant data loss that impedes the prediction results.

B. Quantum optimization for autoregressive models

Note that autoregressive models can be also combined with quantum neural networks as done with the classical neural networks [32]. Also note that autoregressive models can be formulated as combinatorial optimization problems in the framework of graph learning [33]. In addition, it is shown that quadratic unconstrained optimization formulation can be used for forecasting in finance [34]. With the help of these and other similar studies, autoregressive models can be formulated as combinatorial optimization in the form of quadratic unconstrained binary optimization. Therefore, the quantum optimization approaches such as adiabatic quantum computation [35], quantum unconstrained bounded optimization algorithm [36], quantum power iteration [37] can be used for these models.

REFERENCES

- [1] R. P. Feynman, "Quantum mechanical computers," *Foundations of physics*, vol. 16, no. 6, pp. 507–531, 1986.
- [2] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines," *Journal of statistical physics*, vol. 22, no. 5, pp. 563–591, 1980.
- [3] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994, pp. 124–134.

- [4] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Physical review letters*, vol. 79, no. 2, p. 325, 1997.
- [5] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum principal component analysis," *Nature Physics*, vol. 10, no. 9, pp. 631–633, 2014.
- [6] A. N. Kolmogorov, "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition," in *Doklady Akademii Nauk*, vol. 114, no. 5. Russian Academy of Sciences, 1957, pp. 953–956.
- [7] R. Hecht-Nielsen, "Kolmogorov's mapping neural network existence theorem," in *Proceedings of the international conference on Neural Networks*, vol. 3. IEEE Press New York, NY, USA, 1987, pp. 11–14.
- [8] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.
- [9] A. R. Gallant and H. White, "There exists a neural network that does not make avoidable mistakes," in *ICNN*, 1988, pp. 657–664.
- [10] Z.-Q. John Xu, Y. Zhang, T. Luo, Y. Xiao, and Z. Ma, "Frequency principle: Fourier analysis sheds light on deep neural networks," *Communications in Computational Physics*, vol. 28, no. 5, pp. 1746–1767, 2020. [Online]. Available: http://global-sci.org/intro/article_detail/cicp/18395.html
- [11] C. C. Aggarwal *et al.*, *Data mining: the textbook*. Springer, 2015, vol. 1.
- [12] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, vol. 33, no. 4, pp. 917–963, 2019.
- [13] A. Abanda, U. Mori, and J. A. Lozano, "A review on distance based time series classification," *Data Mining and Knowledge Discovery*, vol. 33, no. 2, pp. 378–412, 2019.
- [14] M. S. Gashler and S. C. Ashmore, "Modeling time series data with deep fourier neural networks," *Neurocomputing*, vol. 188, pp. 3–11, 2016.
- [15] V. Dunjko and P. Wittek, "A non-review of quantum machine learning: trends and explorations," *Quantum Views*, vol. 4, p. 32, 2020.
- [16] V. Dunjko and H. J. Briegel, "Machine learning & artificial intelligence in the quantum domain: a review of recent progress," *Reports on Progress in Physics*, vol. 81, no. 7, p. 074001, 2018.
- [17] P. Singh, G. Dhiman, and A. Kaur, "A quantum approach for time series data based on graph and schrödinger equations methods," *Modern Physics Letters A*, vol. 33, no. 35, p. 1850208, 2018.
- [18] D. Emmanoulopoulos and S. Dimoska, "Quantum machine learning in finance: Time series forecasting," *arXiv preprint arXiv:2202.00599*, 2022.
- [19] M. Schuld, R. Sweke, and J. J. Meyer, "Effect of data encoding on the expressive power of variational quantum-machine-learning models," *Physical Review A*, vol. 103, no. 3, p. 032430, 2021.
- [20] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd, "Continuous-variable quantum neural networks," *Physical Review Research*, vol. 1, no. 3, p. 033063, 2019.
- [21] A. Daskin, "A simple quantum neural net with a periodic activation function," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2018, pp. 2887–2891.
- [22] V. Giovannetti, S. Lloyd, and L. Maccone, "Quantum random access memory," *Physical review letters*, vol. 100, no. 16, p. 160501, 2008.
- [23] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.
- [24] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, "Data re-uploading for a universal quantum classifier," *Quantum*, vol. 4, p. 226, 2020.
- [25] M. Schuld and N. Killoran, "Quantum machine learning in feature hilbert spaces," *Physical review letters*, vol. 122, no. 4, p. 040504, 2019.
- [26] E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. G. Green, and S. Severini, "Hierarchical quantum classifiers," *npj Quantum Information*, vol. 4, no. 1, pp. 1–8, 2018.
- [27] H. Buhrman, R. Cleve, J. Watrous, and R. De Wolf, "Quantum fingerprinting," *Physical Review Letters*, vol. 87, no. 16, p. 167902, 2001.
- [28] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, "Exponential improvement in precision for simulating sparse hamiltonians," in *Forum of Mathematics, Sigma*, vol. 5. Cambridge University Press, 2017.
- [29] A. Daskin, A. Grama, G. Kollias, and S. Kais, "Universal programmable quantum circuit schemes to emulate an operator," *The Journal of chemical physics*, vol. 137, no. 23, p. 234112, 2012.
- [30] "Prophet," accessed: Feb. 15, 2022. [Online]. Available: <https://facebook.github.io/prophet/>
- [31] "Qiskit," accessed: Feb. 15, 2022. [Online]. Available: <https://qiskit.org/>
- [32] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.
- [33] Y. Peng, B. Choi, and J. Xu, "Graph learning for combinatorial optimization: A survey of state-of-the-art," *Data Science and Engineering*, vol. 6, no. 2, pp. 119–141, 2021.
- [34] R. Orús, S. Mugel, and E. Lizaso, "Forecasting financial crashes with quantum computing," *Physical Review A*, vol. 99, no. 6, p. 060301, 2019.
- [35] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, "Quantum computation by adiabatic evolution," *arXiv preprint quant-ph/0001106*, 2000.
- [36] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," *arXiv preprint arXiv:1411.4028*, 2014.
- [37] A. Daskin, "Combinatorial optimization through variational quantum power method," *Quantum Information Processing*, vol. 20, no. 10, pp. 1–14, 2021.