Quantum Machine Intelligence (2021) 3: 12
https://doi.org/10.1007/s42484-021-00042-0

RESEARCH ARTICLE

Semi-supervised time series classification method for quantum computing

Sheir Yarkoni[1,2] · Andrii Kleshchonok[1] · Yury Dzerin[1] · Florian Neukart[2,3] · Marc Hilbert[1]

Received: 23 June 2020 / Accepted: 23 February 2021 / Published online: 6 April 2021

Abstract

In this paper we develop methods to solve two problems related to time series (TS) analysis using quantum computing: reconstruction and classification. We formulate the task of reconstructing a given TS from a training set of data as an unconstrained binary optimization (QUBO) problem, which can be solved by both quantum annealers and gate-model quantum processors. We accomplish this by discretizing the TS and converting the reconstruction to a set cover problem, allowing us to perform a one-versus-all method of reconstruction. Using the solution to the reconstruction problem, we show how to extend this method to perform semi-supervised classification of TS data. We present results indicating our method is competitive with current semi- and unsupervised classification techniques, but using less data than classical techniques.

Keywords Quantum computing · Quantum annealing · Quantum machine learning · Classification

1 Introduction

The field of quantum computing has experienced rapid growth in recent years, both in the number of quantum computing hardware providers and their respective processors' computing power. Companies such as D-Wave Systems, Rigetti, and IBM offer access to their quantum processors, and their use in proof-of-concept demonstrations has been widely discussed in literature. Quantum processing units (QPUs) have been used to solve a wide variety of problems such as traffic flow (Neukart et al. 2017), logistics and scheduling (Venturelli et al. 2015; Stollenwerk et al. 2020), quantum simulation (Streif et al. 2019; McCaskey et al. 2019; Grimsley et al. 2019), and more (Venturelli and Kondratyev 2019; Nishimura et al. 2019). Notably, a recent study by Google showed how their QPU can perform the task of sampling from random quantum circuits faster than state-of-the-art classical software (Arute et al. 2019), ushering a new era in the field of quantum computing. These applications use so-called noisy intermediate scale quantum (NISQ; Preskill 2018) processors to solve various forms of optimization and sampling problems. Most commonly, the problem is formulated as a quadratic unconstrained binary optimization (QUBO) problem, or its equivalent form of an Ising Hamiltonian. The former uses a basis of binary $\{0, 1\}$ variables, and the latter makes use of spin variables $\{-1, 1\}$. Both can be solved using existing quantum computing hardware.

The QPUs provided by D-Wave Systems use a quantum annealing algorithm that implements a transverse-field Ising Hamiltonian (Johnson et al. 2011). This quantum protocol prepares an initial Hamiltonian with a simple ground state, and transitions to a Hamiltonian whose ground state is difficult to find. This is referred to as Adiabatic Quantum Computation (AQC) (Van Dam et al. 2001), and under open quantum system conditions as quantum annealing (Kadowaki and Nishimori 1998). Because AQC has been shown to be polynomially equivalent to gate-based quantum computation (Aharonov et al. 2008), and the Ising spin-glass has been shown to be NP-hard to minimize (Barahona 1982), AQC (and quantum annealing) has the potential to significantly impact the fields of optimization, machine learning, and operations research. Equivalently, with gate-model QPUs such as those produced by Google, IBM, and Rigetti, the quantum approximate

Sheir Yarkoni and Andrii Kleshchonok contributed equally to this work.

✉ Sheir Yarkoni
sheir.yarkoni@volkswagen.de

[1] Volkswagen Data:Lab, Ungererstraße 69, Munich, Germany

[2] LIACS, Leiden University, Niels Bohrweg 1, 2333 CA, Leiden, Netherlands

[3] Volkswagen Group of America, San Francisco, CA, USA

optimization algorithm (QAOA) is used to solve such Ising Hamiltonians. This algorithm also attempts to minimize a target Ising Hamiltonian by alternating between a driver and mixer Hamiltonian, until the sampling procedure converges to the target state population. The derivation and details of the QAOA algorithm are beyond the scope of this paper, and are discussed in detail in Farhi et al. (2014).

At the end of a quantum annealing run, or a QAOA circuit execution, the measurements are a projection along the z-component of the qubits' spins, resulting in a sequence of classical bit strings. These states can be interpreted as approximations to finite-temperature Boltzmann states from the classical spin-glass Ising Hamiltonian (Raymond et al. 2016; Verdon et al. 2017):

$$H(s) = \sum_i h_i s_i + \sum_{i<j} J_{ij} s_i s_j. \tag{1}$$

The task of programming a quantum annealer or QAOA circuit involves finding a suitable Ising Hamiltonian representation for the optimization task. In this paper, we motivate classification of time series data based on extracting features which exist within the data, and use combinatorial optimization techniques to match and reconstruct data with other time series. We start by reducing the dimensionality of the TS data and encode it as a string. We introduce a pulling procedure, comparing the encoded strings to form a collection of sets, where common features between the strings are extracted. All extracted common features are pooled together, which we can then use to construct new TS and compare between existing ones. We perform these tasks by using the pulled features as elements of the universe in the set cover problem, which has a known QUBO/Ising formulation and can be solved using quantum computers. By reformulating the critical task in our clustering algorithm as a set cover problem, we introduce two novel ideas to quantum clustering algorithms: (1) we avoid representing single vectors with polynomial numbers of qubits, instead representing the features within the data as the qubits, and (2) we perform the clustering task by transferring the core concepts of clustering (and reconstruction) to the quantum algorithm for set cover, as opposed to a direct translation of a distance-based minimization procedure. This results in an algorithm that avoids a classical "learning" procedure, therefore requiring significantly fewer computational resources compared to other classical and quantum methods.

The rest of this paper is organized as follows. Section 2 provides a short overview of existing methods for both classical and quantum clustering. Section 3 motivates the task of TS reconstruction, explains the methods used to discretize the data, and how to convert the discretized data to the set cover problem and its representative QUBO. Section 4 shows how to extend the reconstruction method to classify the TS data. Section 5 outlines the experimental setup used in this analysis to test the developed method using various open-source data sets, and Section 6 reviews the results from those experiments. Section 7 presents the conclusions from our work and outlines future research in this area.

2 Previous works

Quantum computing-based approaches which exist in literature involve fundamentally different approaches than that introduced in this work; we provide a brief overview of some key methods and algorithms related to quantum clustering. Assuming the existence of error-corrected quantum processors (and the existence of quantum RAM), it has been shown that quantum computers could perform k-means clustering exponentially faster than their classical counterparts (Lloyd et al. 2013). Other works have also shown how to reformulate parts of classical clustering algorithms as quantum subroutines that can be executed on error-corrected gate-model QPUs (Alexander et al. 2018; Aimeur et al. 2013; Wiebe et al. 2015; Horn and Gottlieb 2001). In quantum annealing, a similar approach has been shown in which the objective function of the clustering task (minimizing distance metrics between high-dimensional vectors) has been directly translated to a QUBO, with each vector's possible assignment represented via one-hot encoding to physical qubits (Kumar et al. 2018; Neukart et al. 2018).

Classical time series (TS) analysis is considered to be a challenging task due to high number of dimensions involved resulting in the Curse of Dimensionality phenomenon. A series of works address the question of efficient dimensionality reduction (Keogh et al. 2005; Lin et al. 2003; Lin et al. 2002; Senin et al. 2018; Schäfer and Högqvist 2012; Patel et al. 2002; Guo et al. 2010; Xiaodong et al. 2002), explaining the trade-off between information loss and search space size. Main results presented in this manuscript are obtained with Symbolic Fourier Approximation (SFA) method (Schäfer and Högqvist 2012) due to its pruning power, noise-robustness and scalability. SFA represents each real-valued TS in a frequency domain by a symbolic string using the discrete Fourier transform. These transformed TS can then be used by classical string-based similarity algorithms such as phonetic distance based, Levenshtein, Hamming, Jaro, Jaro-Winkler measures, and more (Gomaa et al. 2013).

Classical TS clustering techniques can be split into the following categories: model-based, feature-based, shape-based and their combinations (Aghabozorgi et al. 2015). In the model-based approach the TS is encoded and fit by parametric models and clustering is applied to these

extracted parameters (Liao 2005). In feature-based methods, the features of TS, like Fourier components, periodicity, trend, number of peaks, and variance, are extracted and later clustered by conventional algorithms (Hautamaki et al. 2008; Christ et al. 2018). Shape-based approaches refer to comparing shapes of TS directly and matching them according to specifically chosen metrics. A typical example for this approach is Dynamic Time Warping (DTW) (Sakoe and Chiba 1978), which has been shown to outperform Euclidean metrics (Chu et al. 2002; Vlachos et al. 2002). DTW-based classical methods are used to evaluate the accuracy of our approach in Section 6. For more details on classical approaches to TS clustering, we refer the reader to Gonzalez et al. (2014), Fu (2011), and Aghabozorgi et al. (2015).

3 Time series reconstruction: problem formulation

Clustering techniques generally require specific data representation, similarity measure definitions, and clustering algorithm selection. Similarly, in our quantum computing-based approach, we represent the TS data as encoded strings from which we formulate semi-supervised clustering and optimal reconstruction as a set cover problem, and provide metrics based on solutions to the set cover problem. While different than classical approaches (Iwama et al. 2018; Acharya et al. 2010; Frieze et al. 1999; Skiena and Sundaram 1995), we preserve the computational complexity of the problem, while introducing a method that is based on latent features within the data.

In order to reconstruct given time series data, we start by discretizing the data, and comparing the encoded strings to generate the elements of our universe to form the set cover. This pulling technique is crucial to allow feature-wise comparison of the data, as well as arbitrary reconstruction of TS using existing (or training) data. We use existing techniques for discretization, and explain the pulling procedure in detail. We then show how to use this data to construct the set cover problem for quantum optimization.

3.1 Discretization and pulling technique

There are many ways to discretize time series data, as reviewed previously. For our purposes, we use the symbolic Fourier approximation (SFA) method (Schäfer and Högqvist 2012), as it provides differentiation between separate TS classes and features in high-dimensional data sets, allowing us to use these representative symbols for our quantum algorithm. Nevertheless, the exact discretization is data-dependent, with various hyperparameters (such as number of letters in the alphabet and length of each encoded string) present in the method; for a full explanation we refer the reader to Schäfer and Högqvist (2012). Given the encoded strings, we compare the time series using the following pulling procedure, illustrated in Fig. 1. This pair-wise comparison is considered a preprocessing step necessary to formulate our set cover problem. Starting with one fixed string (red in the figure), we consider each encoded character as an independent element in the universe set[1] ($U = \{0, 1, 2, 3, 4\}$ in the figure). A second string (green in the figure) is compared element-wise by successively moving the second string along the first, as illustrated. At every iteration, all character matches between the two strings are recorded as a new set. In the example from Fig. 1, the set of sets is $V = \{\{0\}, \{\emptyset\}, \{0, 2\}, \{\emptyset\}, \{1, 2, 3\}, \{\emptyset\}, \{\emptyset\}, \{3\}, \{\emptyset\}\}$.

The procedure is repeated for the rest of the encoded training TS to form the set of sets V. This set, which is a union of all subsets obtained via the pulling technique, represents the features in common between the target time series and all other time series in the data set. Given this aggregate set, the goal is now to select the minimal subset that most closely reconstructs the universe, which is the NP-hard set cover problem. In the case illustrated in Fig. 1, the optimal selection of subsets is underlined in red. In principle, solutions of this set cover problem do not preserve order of elements, and allow the use of the same element multiple times. This feature is useful for TS comparison, as elements of the time series data can be permuted and duplicated without affecting our reconstruction method.

3.2 Formulating the set cover problem

Given the encoded strings and the common set of features V, we can now formulate the set cover problem as a QUBO, following the method demonstrated in Lucas (2014). Consider the universe set $U = \{1, ..., n\}$, and a set of subsets V_i, such that $U = \bigcup_{i=1}^{N} V_i$, $V_i \subseteq U$. Finding the smallest number of subsets V_i whose union is U is a well-known NP-hard optimization problem in the worst case (Karp 1972). In order to map the set cover problem to a QUBO problem, we use the following binary variables:

$$x_i = \begin{cases} 1, & \text{if set } i \text{ is included,} \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

and

$$x_{\alpha,m} = \begin{cases} 1, & \text{if the number of } V_i \text{ which include element } \alpha \text{ is equal to } m, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

[1] It is important to note that by using the same encoding scheme for all TS data, we ensure that all string characters belong to the same alphabet.

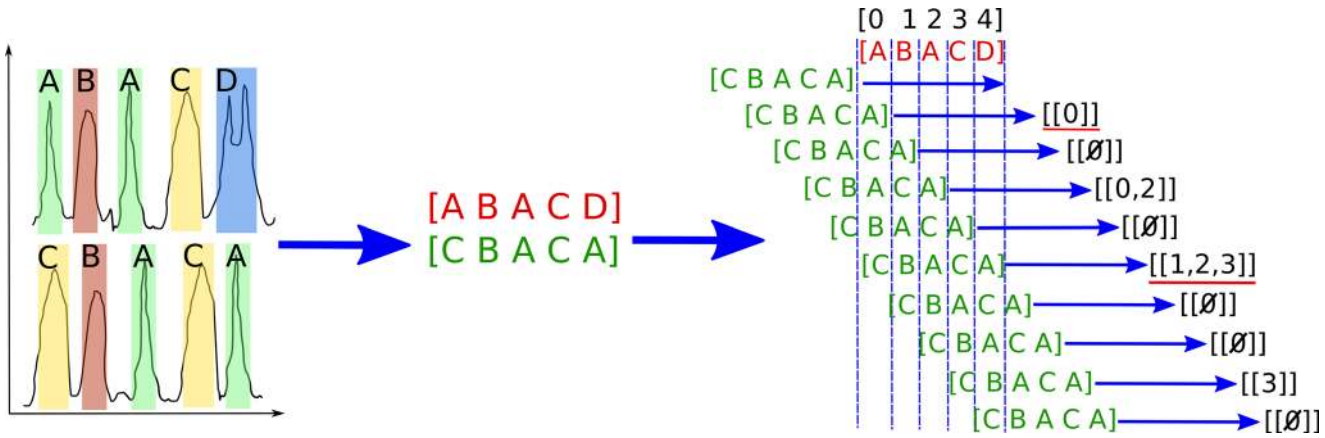

Fig. 1 Schematic illustration of TS encoding and pulling procedure to produce subsets of set $V = \{\{0\}, \{\emptyset\}, \{0, 2\}, \{\emptyset\}, \{1, 2, 3\}, \{\emptyset\}, \{\emptyset\}, \{3\}, \{\emptyset\}\}$. The optimal selection to cover $U = \{0, 1, 2, 3, 4\}$ in this case would be underlined subsets $V = \{\{0\}, \{1, 2, 3\}\}$ with item numbers 0 and 4

Here, $\alpha \in U$ denotes an element of universe set, and m signifies if element α appears in m subsets. We consider the full QUBO as a sum of two components:

$$H_A = A \sum_{\alpha=1}^{n} \left(1 - \sum_{m=1}^{N} x_{\alpha,m}\right)^2 + A \sum_{\alpha=1}^{n} \left(\sum_{m=1}^{N} m x_{\alpha,m} - \sum_{i:\alpha \in V_i} x_i\right)^2, \tag{4}$$

and

$$H_B = B \sum_{i=1}^{N} x_i. \tag{5}$$

The complete QUBO is given by $H = H_A + H_B$ (Lucas 2014). The first summation in H_A imposes that exactly one of $x_{\alpha,m}$ must be selected in the minimum via a one-hot encoding. The second term in H_A represents the number of times α is selected, and that this is equal to the number of selected subsets α appears in (m, as only one $x_{\alpha,m}$ can be 1 in the minimum). The final term H_B (5) serves to minimize the number of V_i needed to cover the universe U. The total number of variables required is $N + n(1 + M)$, where M is the maximal number of sets that contain given element of U (see Lucas (2014) for details). The limiting case where each element of V_i included covers only one element of U constrains the coefficient of H_A and H_B to $0 < B < A$. The closer the coefficient B and A, the more weight is given to (5), minimizing the number of elements selected from V.

In our application of time series reconstruction, the final size of the QUBO is heavily dependent on our choices during discretization. For example, the number of binary variables is equal to $N_{\text{train TS}}\,(2L - 1)\,(L + 1)$, where $N_{\text{train TS}}$ is the number of TS in the training set used for reconstruction, and L is the length of string that encodes the TS. Increasing the string length to encode each TS changes the size of the universe U. Allowing longer encoded strings to represent the data creates more subsets V_i. Therefore, there exists a trade-off between the granularity of the encoded strings and the ability to solve the set cover representation of the problem. Including more characters in our alphabet for discretization changes the non-empty sets V_i, which the number of quadratic elements in the QUBO depends on. The general trend is, however, that the number of the quadratic element decreases with the increase of the characters used in our alphabet. This is explained by the properties of the pulling procedure described above, since a smaller alphabet produces more non-empty elements V_i which could be used for reconstruction of the universe U. In Fig. 2 we show how varying these hyperparameters of the discretization affects the size of the QUBO problem, based on 20 test samples from the BeetleFly data set (Hills et al. 2014).

4 Semi-supervised classification

We can now combine the methods described in the previous sections—constructing the universe set U from discretized data and the subsets V—to perform semi-supervised clustering. We start by separating the input TS data into two groups—training and test data. In our case we use training data sets with known labels, and the task we solve is to use the labeled data to assign labels to the test set. Normally, the training set with labeled data is significantly smaller than unlabeled test set, which we exploit in our method.

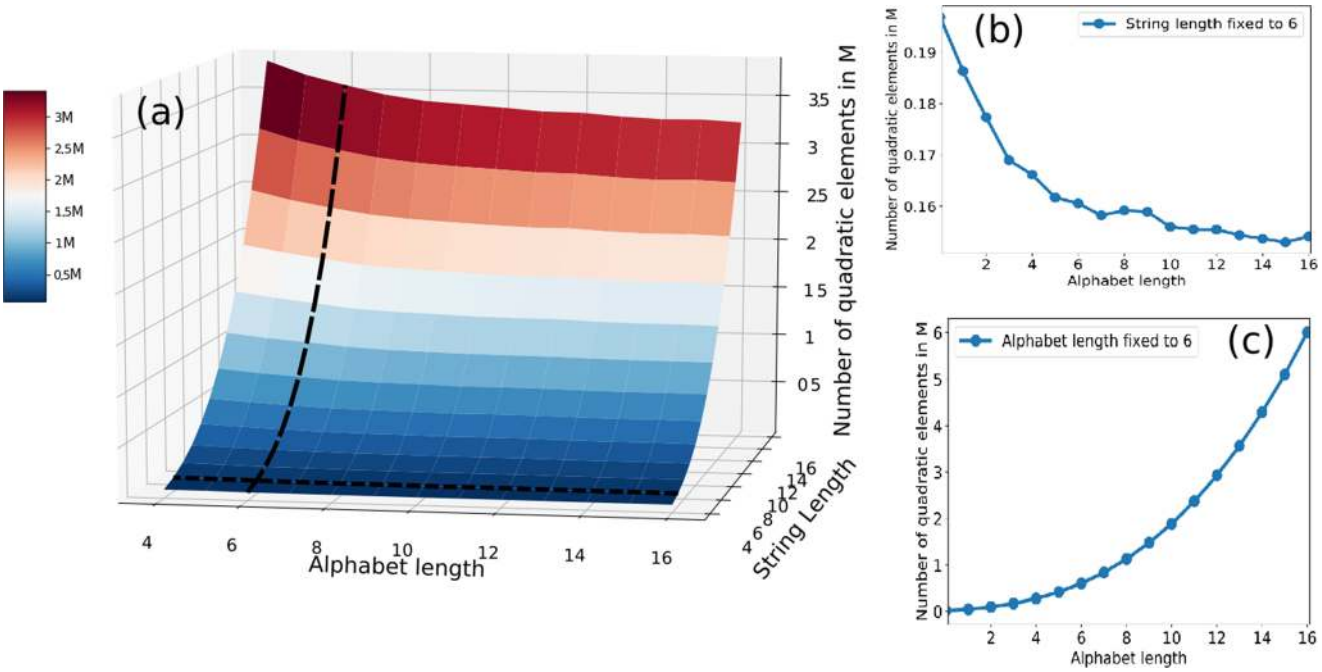

Fig. 2 **a** The number of quadratic terms in millions as a function of string and alphabet length. **b** Quadratic elements as a function of alphabet length, with string length being fixed to 6. **c** Quadratic terms as function of string length, with alphabet length being fixed to 6. The corresponding isolines (**b**, **c**) are shown with dashed line on surface plot (**a**). Analysis was performed using 20 test samples from the BeetleFly data set (Hills et al. 2014)

We encode both the training and test data sets into strings using the method described in Section 3.1. We then perform the reconstruction procedure for every TS in our test set using the entire training set. Each TS from the test set is assumed to individually form a universe U, and is to be reconstructed using the sets V_i, obtained via the pulling procedure. Explicitly, using Fig. 1, the red string is the TS from the test data set, and all strings in the training set are pulled through (green strings) to obtain the V_i's. This allows us to compare every test TS to the full training set in one-versus-all manner. Then, using the universe U and V_i's from the pulling procedure, we formulate the set cover problem outlined in Section 3.2. Thus, a single solution to that set cover problem (even sub-optimal in the worst case) allows us to reconstruct each TS from the test set using a set of discretized features obtained from *all* elements which appear in the training set. Furthermore, since annealing-based sampling methods produce finite-temperature Boltzmann distributions (Raymond et al. 2016), various optima of the set cover problem could yield different ways to reconstruct the test TS using the training set. Due to this, it is therefore the users' task to use these reconstructions to associate each test TS with a label from the training set. We outline the steps of our classification procedure using pseudo-code in Algorithm 1.

Algorithm 1 Semi-supervised QUBO-based clustering algorithm.

DIMENSIONALITY REDUCTION
Apply encoding of all TS into a string of characters using SFA (Schäfer and Högqvist 2012) or PAA (Guo et al. 2010)-like methods.
for TS in test data **do**
PRODUCE SET OF SETS FOR COVER PROBLEM
Assume encoded TS forms the universe U. Apply pulling procedure (Fig. 1) to the full training data set to compute V and map the problem to the set cover problem.
FORMULATE THE QUBO PROBLEM
Generate set cover QUBO an according to Section 3.
QUBO OPTIMIZATION
Find minimum of the QUBO; record subsets V_i selected to cover U.
ANALYZE RESULTS OF QUBO OPTIMIZATION
Trace back selected V_i to respective time series from training set, and assign label to TS based on selection metrics.
end for

To classify the reconstructed test TS data we evaluated three different similarity metrics using set cover solutions:

largest common subset V_i, highest number of common subsets V_i, and largest sum of common elements in selected V_i. We briefly explain how each metric is calculated, and discuss the performance of each.

- **Largest common subset.** Given a candidate solution to the set cover problem, the label corresponding to the V_i which contains the most elements is selected. The label is then assigned to the test TS. This metric captures the longest continuous set of features from the training TS data, and assumes that is sufficient to determine the label.
- **Number of common subsets.** Frequently, multiple V_i's from the same training TS are used to reconstruct a test TS. In this metric, we count the number of V_i subsets used to cover the universe. The test label is assigned the same label as the training TS which appears most frequently in the set cover solution.
- **Largest sum of subsets.** This metric is a combination of the previous two. For every training TS that is used to reconstruct a test set, the total number of elements used by each is counted (summed over all V_i's). The label which corresponds to the training TS with the largest sum is assigned to the test TS.

These metrics allow us to quantify the accuracy of our semi-supervised clustering algorithm. The first two metrics, being based on large sets of common features between the TS, performed the best (results shown in the next section). There was no significant difference between the two metrics, and the superiority of one metric over the other varied between data sets. The third metric, which was a combination of the first two, performed worse than either of the first metrics in the majority of the cases tested. While unexpected to begin with, this observation could be explained by the fact that because the third metric admits matches with many small subsets V_i that are selected in the set cover, this metric could miss significant signatures present in the TS data. The largest common subset metric was selected for the experiments presented in the next section. It should also be noted that the use of labeled training data is not designed to not reach the accuracy of supervised learning methods. Moreover, there are modifications that could be made to the methods presented to improve the accuracy, for example increasing the word length and/or using a larger train set. Both are constrained in our use-case to prohibit excessively large QUBOs from being constructed. The goal of this method, as described, is to allow for relatively high accuracy using small sets of training data.

We provide an illustrative example of our QUBO-based reconstruction and classification in Fig. 3 using the BeetleFly data set. The task is to reconstruct the data in Fig. 3a using (b) and (c). For this example, an alphabet of size 5 was used for encoding, color-coded in the figure. The results of the set cover problem, formulated using the methods explained in previous sections, are three sets, shown as v_1, v_2, and v_3 in Fig. 3. Meaning, each box (representing a fifth of the TS data per box) that appears in one of the subsets forming the solution is designated as such. Specifically, v_1 = ['A', 'E'], v_2 = ['E', 'B'], and v_3 = ['C']. Therefore, the union $v_1 \bigcup v_2 \bigcup v_3 = U$, where U ='ACEEB', the test TS data to reconstruct. For classifying the reconstructed sample, we refer to the classes of the training data used for the reconstruction, and note that the training samples in Fig. 3b and c belong to two different classes. Using the similarity metrics defined above, it is easy to determine that v_1 and v_2 both originate from the time series (b), whereas only v_3 (which contains only a single element) is obtained from (c). Therefore, (a) is assigned the same label as (b). This example is representative of the majority of cases encountered during classification, with components of the reconstructed TS varying across multiple training samples, and often also across multiple classes.

5 Experimental setup

The experiments performed in this work used open-source labeled TS data available publicly (Bagnall et al. 2017; Bagnall et al.). We restricted our analysis to univariate TS data with two classes and small training set size to make our work amenable to NISQ devices in the near future. However, this method of semi-supervised classification can be used with any number of classes, at the cost of QUBO size. Since both the number of TS in the training data and the word length used to encode the TS contribute to the number of variables in the QUBO, we select data sets that have small numbers of TS in the training set. The test and training sets used in these experiments are already determined and labeled by the source, allowing us to easily calculate the classification rate of our method and avoid the step of selecting a training set. To benchmark the performance of our classification method, we compared the accuracy of our labeling to semi-supervised and unsupervised classical classification methods. The results of these experiments for the various data sets are summarized in Table 2.

5.1 Data sources

To test the robustness of our method we collected a variety of data sources of different types. We briefly review each source and provide a literature reference for further details. We note that in the data sources' accompanying cited works, higher classification rates than our methods are reported using supervised algorithms. In our analysis we do not consider supervised classification algorithms, and compare

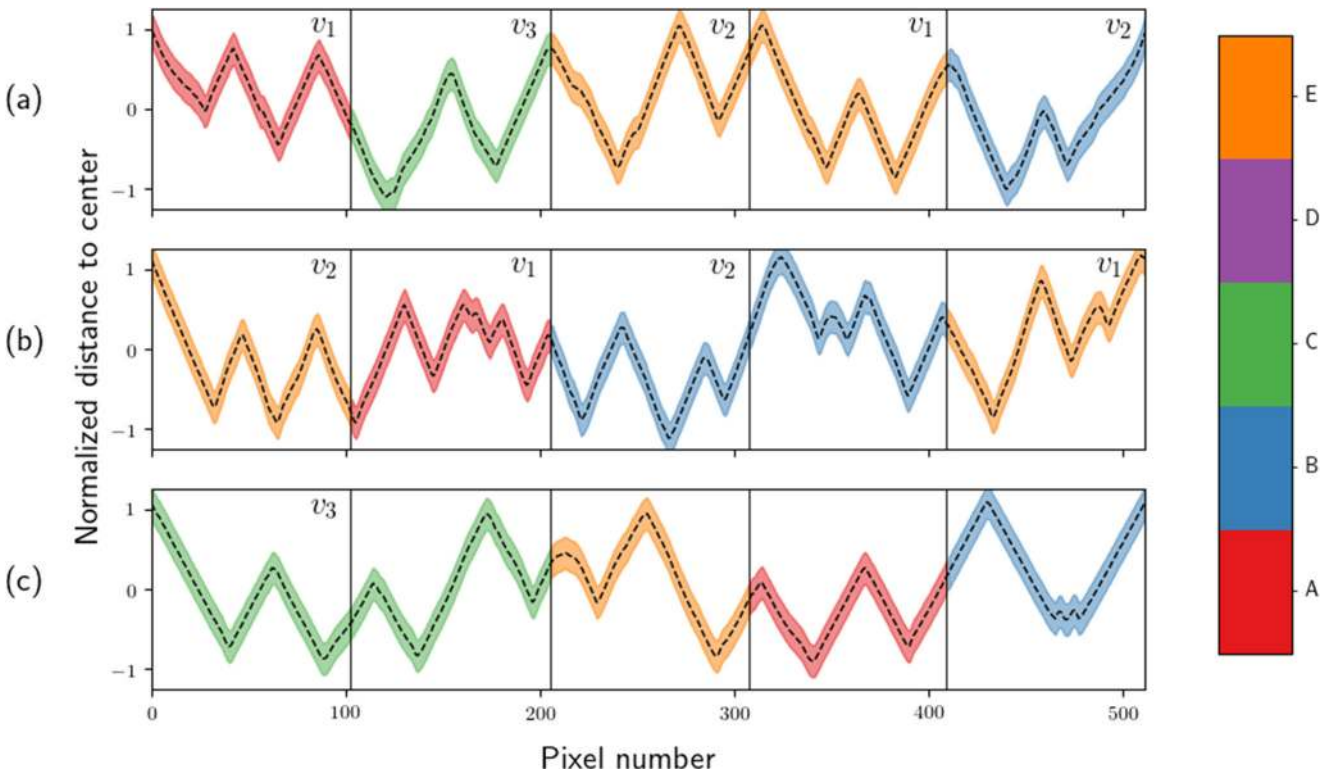

Fig. 3 An illustrative example of reconstruction and classification from the BeetleFly data set. **a** A test TS sample (encoded as 'ACEEB') reconstructed from two training TS. Each box in the sub-figure is encoded as a single letter in a string, as per the color bar. The subsets v_i obtained from the pulling procedure and used to reconstruct this data are shown both in the reconstructed (test) TS and in the training TS. **b** The first training data used for reconstruction and classification (encoded as 'EABBE'). **c** A second time series used for reconstruction (encoded as 'CCEAB')

our semi-supervised quantum-based approach to similar classical algorithms.

SonyAIBORobotSurface1 (Mueen et al. 2011) data is sensor data collected from a small, dog-shaped, quadruped robot. It is equipped with multiple sensors, including a tri-axial accelerometer. In the experiments section we classify between roll accelerometer measurements on two classes of surfaces: soft carpet and hard cement.

GunPoint (Ratanamahatana and Keogh 2005) data includes motion tracking of actors' hands during gun-drawing and gun-pointing actions. For both classes the X-component of the actor's right hand centroid is tracked and used to distinguish between the two classes.

TwoLeadECG (AL et al. 2003) and **ECG200** (AL et al. 2003) are electrocardiogram data sets available at the PhysioNet database (AL et al. 2003; Wagner et al. 2020). The first includes long-term measurements from the same patient using two different leads. The classification task aims to differentiate between each lead signal. In contrast, the second ECG200 (AL et al. 2003) set contains electrical activity recorded during one heartbeat. The two classes are the normal heartbeat and a Myocardial Infarction records.

BeetleFly (Hills et al. 2014) time series data is generated from binary images developed for the testing of shape descriptors. The external contour of these images is extracted and mapped into the distance to the image center. The two image classes are contours of beetles and flies.

Chinatown (Chinatown 2020) data is collected by an automated pedestrian counting system in the city of Melbourne, Australia. The classes are based on weekday or weekend traffic.

It is important to note that all data sources used in this experiment are real-world data sources which are available for public use. Furthermore, the sources are pre-divided into labeled training and test sets. These sets were used as-is in the experiments performed below. Validation is performed by measuring the classification rates of each methods on the labeled test data.

5.2 QUBO sizes and optimization

The QUBOs generated by our methods were too large to be optimized using the largest available QPUs (D-Wave 2000Q) at the time of experiments. The exact sizes of the QUBOs for each data set are shown in Fig. 4. To solve the QUBOs we used simulated thermal annealing (SA), a well-known classical heuristic for solving such optimization problems. The specific implementation of SA was from the

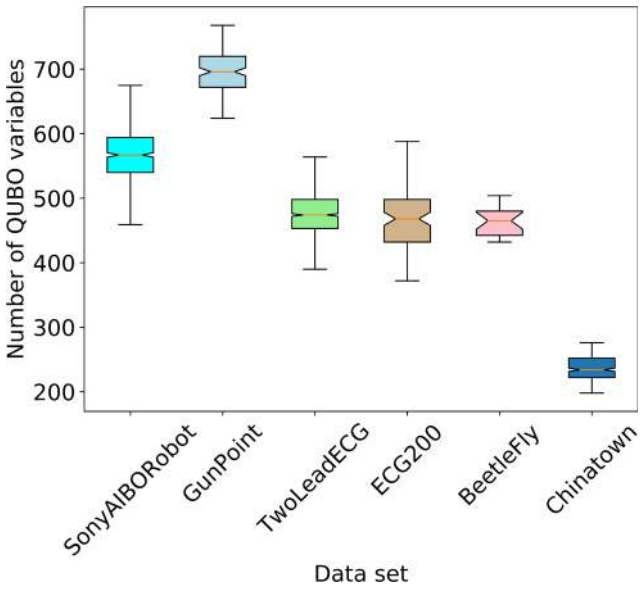

Fig. 4 Distribution of number of QUBO variables for all data sets in Table 1

D-Wave Python package for classical QUBO optimizers (D-Wave Systems 2021). We found that 20,000 samples and 1000 SA sweeps (with geometric interpolation of the inverse temperature) were sufficient to ensure that low-energy local minima were sampled within reasonable times per QUBO. We use the default SA settings in the package for initial and terminal inverse temperature selection (for more information about the implementation of SA we refer the reader to (D-Wave Systems 2021)).

The specific parameters used for the TS encoding to generate the QUBOs are shown in Table 1. In general, the longer the TS are and the fewer TS are in the training set, the finer the discretization method required to accurately classify the test data. In all data sets we were able to reconstruct each test TS with elements from the training set, as explained in Section 3.2. The distributions of the number of variables in each QUBO for all data sets is shown in Fig. 4.

5.3 Classification benchmarking

For the purposes of evaluating our QUBO-based classification method quantitatively, we compare two classical time series classification algorithms based on dynamical time warping (DTW) (Sakoe and Chiba 1978) measures: k-means clustering and a classical analogue of the semi-supervised method described in the paper. The motivation for using these specifically is that both are based on pairwise similarity metrics as in our approach. DTW applied to temporal sequences aligns the pair series in a non-linear way to minimize differences and calculate Euclidean distance afterwards. The DTW measure could be applied directly in unsupervised k-means clustering or similarly to the method described in our paper in the semi-supervised fashion We use k-means clustering with pairwise DTW metrics calculated on the original TS (before encoding), with the labels being assigned based on belonging to one of two clusters. The second method assigns the test TS labels are by the DTW metric directly, calculated pairwise between each training and test TS (without encoding). We use these two methods to calculate classification rates for all data sources in the experiments (Table 2).

6 Results

As expected, the semi-supervised QUBO-based method outperforms classical unsupervised methods. We note however, that the QUBO-based method operates on a reduced dimensionality in contrast to the classical methods which use the original TS, where full information is preserved. Even under this consideration the accuracy of QUBO-based method is comparable with the semi-supervised DTW methods, and could be improved still by enriching the set V, i.e. by augmenting the training set or increasing the discretization granularity.

The worst performance of the QUBO-based algorithm is observed on the TwoLeadECG data set. This could be explained by the nature of our method, as well as the

Table 1 Table with data set description, number of TS in training and test sets, length of TS, and length of each encoded string and number of different letters used to encode data set

Data set	Data type	Train/test size	Time series length	Word/alphabet length
SonyAIBORobotSurface1 (Mueen et al. 2011)	Sensor	10/601	70	8/8
GunPoint (Ratanamahatana and Keogh 2005)	Motion	30/150	150	5/5
TwoLeadECG (AL et al. 2003)	ECG	20/1139	82	5/5
ECG200 (AL et al. 2003)	ECG	20/100	96	5/5
BeetleFly (Hills et al. 2014)	Image	20/20	512	5/5
Chinatown (Chinatown 2020)	Traffic	20/345	24	5/5

Table 2 The clustering accuracy measured on two classes and weighted average reported for QUBO-based and classical DTW-based methods

Data set	QUBO method class1/class2/weighted	K-means clustering class1/class2/weighted	DTW semi-supervised class1/class2/weighted
SonyAIBORobotSurface1 (Mueen et al. 2011)	0.7/0.9/0.78	**0.85/0.97/0.92**	0.97/0.63/0.83
GunPoint (Ratanamahatana and Keogh 2005)	**0.76/0.79/0.78***	0.53/0.51/0.52	**0.82/0.77/0.79***
TwoLeadECG (AL et al. 2003)	0.6/0.62/0.61	0.65/0.7/0.68	**0.86/0.94/0.9**
ECG200 (AL et al. 2003)	0.61/0.82/0.75	**0.62/0.8/0.79**	0.87/0.51/0.64
BeetleFly (Hills et al. 2014)	**0.85/0.89/0.87**	0.64/0.83/0.73	0.62/1.0/0.82
Chinatown (Chinatown 2020)	0.72/0.91/0.86	0.37/0.78/0.67	**0.89/0.98/0.94**

Bold text signifies the most effective classification method (based on the weighted average of the two classes) for each data set tested. Asterisk denotes a tie between the methods within statistical variance

sensitivity of the ECG data. By using the set cover problem, we allow for permutations of subsets of TS data in the reconstruction of the test TS. It is likely that this permutation of TS segments, and similar representation in Fourier space of the signals from the two leads in the ECG measurements, makes our method not suitable for this kind of data. To confirm this is the case, and improve the classification accuracy, we applied SAX (Senin et al. 2018) encoding, based on sliding window time series magnitude, rather than the Fourier transform. Using this method, and encoding the TS as a word of length 5 constructed from a 5 letter alphabet, the accuracy is improved to 0.62 and 0.85 for the two respective classes (0.74 weighted average). In contrast to the Fourier encoding, the better results with ECG200 data are due to the significant differences between the classes of normal and ischemia ECG readings.

The highest accuracy is obtained using the BeetleFly and Chinatown data sets. In the first case, many permutations of the training set to construct the test set are permissible, which our method takes advantage of. The accuracy of our method is additionally improved by the relative size of the training set, further augmenting the combinatorial space of permutations. This robustness can also be explained by the dimensionality reduction technique for this data set: the 2D BeetleFly images (with different orientations) were mapped to 1D series of distances to the image centre, which again is beneficial for permutation-based methods. The Chinatown data set, for comparison, contained significantly shorter TS than BeetleFly. Encoding the Chinatown TS data with the same word length as BeetleFly resulted in higher granularity representations, and ultimately higher accuracy. This provides additional evidence that the accuracy of our method can be improved by increasing the granularity of the encoding.

7 Conclusions

We present a QUBO-based method for TS reconstruction and semi-supervised classification that reaches accuracy scores comparable with classical DTW pairwise approaches, and in most cases outperforms unsupervised clustering. Among the advantages of our method is the utilization of significantly less data with respect to conventional classical methods, as well as a one-versus-all comparison that allows the selection of segments of data from multiple sources to reconstruct a single TS. This provides an additional robustness in our method in permutations of TS segments during the reconstruction. We showed how to reformulate the task of TS reconstruction as the set cover problem with a minimal number of subsets. In order to formulate this problem as a QUBO we apply TS dimensionality reduction by encoding each time series as a separate string. This encoding procedure and selection of comparison metrics (as discussed in Section 4) define the hyperparameter space of the problem. The QUBO-based classification method performed the best on image and traffic data, which is consistent with our method's inherit ability to utilize permutations of features/data within the TS to perform reconstruction.

Time series reconstruction and classification has a wide variety of useful applications, such as: management of energy systems, factory process control, sensor systems, and many more. The methods introduced in this paper show how to reformulate the tasks of reconstruction and classification of such data using quantum computing. The fact that our work uses small training sets of labeled data means that the QUBOs produced could be solved by next-generation NISQ devices. Using quantum technologies, this method could analyze significantly more complex TS data, even

in a live setting. The results of the optimization process (the selected subsets used for the reconstruction) would be informative as feedback for live process optimization as well. Future work in this are will be focused on generalising the method to multivariate TS cases, finding application-ready data sets, and execution of the presented methods on quantum processors. Specifically, with the advancement of hybrid quantum-classical algorithms, we will focus on converting the methods presented in this paper to be suitable for commercial applications.

References

Acharya J, Das H, Milenkovic O, Orlitsky A, Pan S (2010) On reconstructing a string from its substring compositions. 1238–1242 07

Aghabozorgi S, Shirkhorshidi AS, Wah TY (2015) Time-series clustering – a decade review. Inf Syst 53:16–38

Aharonov D, Van Dam W, Kempe J, Landau Z, Lloyd S, Regev O (2008) Adiabatic quantum computation is equivalent to standard quantum computation. SIAM Rev 50(4):755–787

Aimeur E, Brassard G, Gambs S (2013) Quantum speed-up for unsupervised learning. Mach Learn 90(02):261–287

AL G, LAN A, L G, JM I, Hausdorff amd PC, RG M, JE M, Moody G, C-K P, Stanley H (2003) Mit-bih long-term ecg database. physionet.org/content/ltdb/1.0.0/. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. Circulation 101(23):e215-e220

Alexander C, Shi L, Akhmametyeva S (2018) Using quantum mechanics to cluster time series. arXiv

Arute F, Arya K, Babbush R, Bacon D, Bardin JC, Barends R, Biswas R, Boixo S, Brandao FG, Buell DA et al (2019) Quantum supremacy using a programmable superconducting processor. Nature 574(7779):505–510

Bagnall A, Lines J, Bostrom A, Large J, Keogh E (2017) The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining Knowl Disc 31(3):606–660

Bagnall A, Lines J, Vickers W, Keogh E The uea & ucr time series classification repository. www.timeseriesclassification.com. Accessed: 2020-02-01

Barahona F (1982) On the computational complexity of ising spin glass models. J Phys A Math Gen 15(10):3241

Chinatown DHA (2020). . http://www.pedestrian.melbourne.vic.gov.au. Accessed: 2020-02-01

Christ M, Braun N, Neuffer J, Kempa-Liehr AW (2018) Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package). Neurocomputing 307:72–77

Chu S, Keogh EJ, Hart DM, Pazzani MJ (2002) Iterative deepening dynamic time warping for time series. In: SDM

D-Wave Systems (2021) D-Wave Dimod Package. https://docs.ocean.dwavesys.com/projects/dimod/

Farhi E, Goldstone J, Gutmann S (2014) A quantum approximate optimization algorithm. arXiv:1411.4028

Frieze AM, Preparata FP, Upfal E (1999) Optimal reconstruction of a sequence from its probes. J Comput Biol 6(3-4):361–368

Fu T (2011) A review on time series data mining. Eng Appl Artif Intell 24(1):164–181

Gomaa WH, Fahmy AA et al (2013) A survey of text similarity approaches. Int J Comput Appl 68(13):13–18

Gonzalez JA, Zolhavarieh S, Aghabozorgi S, Teh YW (2014) A review of subsequence time series clustering. The Scientific World Journal 312521

Grimsley HR, Economou SE, Barnes E, Mayhall NJ (2019) An adaptive variational algorithm for exact molecular simulations on a quantum computer. Nat Commun 10(1):3007

Guo C, Li H, Pan D (2010) An improved piecewise aggregate approximation based on statistical features for time series mining. In: Bi Y, Williams M-A (eds) Knowledge Science, Engineering and Management. Springer, Berlin, pp 234–244

Hautamaki V, Nykanen P, Franti P (2008) Time-series clustering by approximate prototypes. In: 2008 19th International conference on pattern recognition, pp 1–4

Hills J, Lines J, Baranauskas E, Mapp J, Bagnall A (2014) Classification of time series by shapelet transformation. Data Min Knowl Disc 28(4):851–881

Horn D, Gottlieb A (2001) The method of quantum clustering. In: Proceedings of the 14th international conference on neural information processing systems: natural and synthetic, NIPS'01, pp769–776, Cambridge, MA, USA, MIT Press

Iwama K, Teruyama J, Tsuyama S (2018) Reconstructing strings from substrings: Optimal randomized and average-case algorithms. arXiv:1808.00674

Johnson MW, Amin MHS, Gildert S, Lanting T, Hamze F, Dickson N, Harris R, Berkley AJ, Johansson J, Bunyk P, Chapple EM, Enderud C, Hilton JP, Karimi K, Ladizinsky E, Ladizinsky N, Oh T, Perminov I, Rich C, Thom MC, Tolkacheva E, Truncik CJS, Uchaikin S, Wang J, Wilson B, Rose G (2011) Quantum annealing with manufactured spins. Nature 473(7346):194–198

Kadowaki T, Nishimori H (1998) Quantum annealing in the transverse ising model. Phys Rev E 58:5355–5363

Karp R (1972) Reducibility among combinatorial problems, volume 85. Complexity of Computer Computations

Keogh E, Lin J, Fu A (2005) Hot sax: Efficiently finding the most unusual time series subsequence. In: Proceedings of the Fifth IEEE international conference on data mining, ICDM '05, pp 226–233, USA, IEEE Computer Society

Kumar V, Bass G, Tomlin C, Dulny J (2018) Quantum annealing for combinatorial clustering. Quantum Inf Process 17(2):39

Liao TW (2005) Clustering of time series data—a survey. Pattern Recogn 38(11):1857–1874

Lin J, Keogh E, Lonardi S, Chiu B (2003) A symbolic representation of time series, with implications for streaming algorithms. In: Proceedings of the 8th ACM SIGMOD workshop on research issues in data mining and knowledge Discovery, DMKD '03, page 2–11, New York, NY, USA. Association for Computing Machinery

Lin J, Keogh E, Lonardi S, Patel P (2002) Finding motifs in time series. Proceedings of the Second Workshop on Temporal Data Mining 10

Lloyd S, Mohseni M, Rebentrost P (2013) Quantum algorithms for supervised and unsupervised machine learning

Lucas A (2014) Ising formulations of many np problems. Front Phys 2:5

McCaskey AJ, Parks ZP, Jakowski J, Moore SV, Morris TD, Humble TS, Pooser RC (2019) Quantum chemistry as a benchmark for near-term quantum computers. npj Quantum Inform 5(1):99

Mueen A, Keogh E, Young N (2011) Logical-shapelets: an expressive primitive for time series classification. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 1154–1162. ACM

Neukart F, Compostella G, Seidel C, von Dollen D, Yarkoni S, Parney B (2017) Traffic flow optimization using a quantum annealer. Frontiers in ICT 4:29

Neukart F, Dollen DV, Seidel C (2018) Quantum-assisted cluster analysis on a quantum annealing device. Front Phys 6:55

Nishimura N, Tanahashi K, Suganuma K, Miyama MJ, Ohzeki M (2019) Item listing optimization for e-commerce websites based on diversity. Front Comput Sci 1:2

Patel P, Keogh EJ, Lin J, Lonardi S (2002) Mining motifs in massive time series databases. In: 2002 IEEE International conference on data mining, 2002. Proceedings, pp 370–377

Preskill J (2018) Quantum Computing in the NISQ era and beyond. Quantum 2:79

Ratanamahatana CA, Keogh EJ (2005) Three myths about dynamic time warping data mining. In: Kargupta H, Srivastava J, Kamath C, Goodman A (eds) SDM, pp 506–510. SIAM

Raymond J, Yarkoni S, Andriyash E (2016) Global warming: Temperature estimation in annealers. Front ICT 3:23

Sakoe H, Chiba S (1978) Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans Acoust Speech Signal Process 26(1):43–49

Schäfer P, Högqvist M (2012) Sfa: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In: Proceedings of the 15th International conference on extending database technology, pp 516–527. ACM

Senin P, Lin J, Wang X, Oates T, Gandhi S, Boedihardjo AP, Chen C, Frankenstein S. (2018) Grammarviz 3.0: Interactive discovery of variable-length time series patterns. ACM Trans Knowl Discov Data 12(1):10:1–10:28

Skiena SS, Sundaram G (1995) Reconstructing strings from substrings. J Comput Biol 2(2):333–353

Stollenwerk T, O'Gorman B, Venturelli D, Mandrà S, Rodionova O, Ng H, Sridhar B, Rieffel EG, Biswas R (2020) Quantum annealing applied to de-conflicting optimal trajectories for air traffic management. IEEE Trans Intell Transp Syst 21(1):285–297

Streif M, Neukart F, Leib M (2019) Solving quantum chemistry problems with a d-wave quantum annealer. In: Feld S, Linnhoff-Popien C (eds) Quantum technology and optimization problems. Springer International Publishing, Cham, pp 111–122

Van Dam W, Mosca M, Vazirani U (2001) How powerful is adiabatic quantum computation? In: Proceedings 42nd IEEE symposium on foundations of computer science. pp 279–287 IEEE

Venturelli D, Kondratyev A (2019) Reverse quantum annealing approach to portfolio optimization problems. Quantum Mach Intell 1(1):17–30

Venturelli D, Marchand DJ, Rojo G (2015) Quantum annealing implementation of job-shop scheduling. arXiv:1506.08479

Verdon G, Broughton M, Biamonte J (2017) A quantum algorithm to train neural networks using low-depth circuits

Vlachos M, Gunopulos D, Kollios G (2002) Discovering similar multidimensional trajectories. In: Proceedings 18th international conference on data engineering. pp 673–684

Wagner P, Strodthoff N, Bousseljot R-D, Kreiseler D, Lunze FI, Samek W, Schaeffter T (2020) Ptb-xl, a large publicly available electrocardiography dataset. Scient Data 7(1):154

Wiebe N, Kapoor A, Svore K (2015) Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning. Quantum Inform Comput 15:318–358 03

Xiaodong F, Changling C, Changling L, Shao H (2002) An improved process data compression algorithm. In: Proceedings of the 4th world congress on intelligent control and automation (Cat. no.02EX527) volume 3, vol 3, pp 2190–2193