Quantum normalizing flows for anomaly detection

Bodo Rosenhahn and Christoph Hirche
Institute for Information Processing (tnt/L3S), Leibniz Universität Hannover, 30167 Hannover, Germany

(Received 6 February 2024; accepted 29 July 2024; published 28 August 2024)

A normalizing flow computes a bijective mapping from an arbitrary distribution to a predefined (e.g., normal) distribution. Such a flow can be used to address different tasks, e.g., anomaly detection, once such a mapping has been learned. In this work we introduce normalizing flows for quantum architectures, describe how to model and optimize such a flow, and evaluate our method on example datasets. Our proposed models show competitive performance for anomaly detection compared to classical methods, especially those ones where there are already quantum inspired algorithms available. In the experiments we compare our performace to isolation forests (IF), the local outlier factor (LOF), or single-class SVMs.

DOI: 10.1103/PhysRevA.110.022443

I. INTRODUCTION

Anomaly detection is the task to identify data points, entities, or events that fall outside a normal range. Thus an anomaly is a data point that deviates from its expectation or the majority of the observations. Applications are in the domains of cybersecurity [1], medicine [2], machine vision [3], (financial) fraud detection [4], or production [5]. In this work we assume that only normal data are available during training. Such an assumption is valid in production environments where many positive examples are available and events happen rarely which lead to faulty examples. During inference, the model has to differ between normal and anomalous samples. This is also termed semi-supervised anomaly detection [6], novelty detection [7,8], or one-class classification [9].

In this work we will make use of normalizing flows [10] for anomaly detection. A normalizing flow (NF) is a transformation of an arbitrary distribution, e.g., coming from a dataset to a provided probability distribution (e.g., a normal distribution). The deviation from an expected normal distribution can then be used as anomaly score for anomaly detection. A defining property of normalizing flows is the bijectivity, thus an NF can be evaluated as a forward and backward path, an aspect which is trivial for quantum gates which can be represented as unitary matrices. Another aspect is that on a quantum computer the output is always a distribution of measurements. This distribution can be directly compared to the target distribution by using a KL-divergence measure for evaluation. In general, this step will require sampling. We would like to raise two aspects as to why quantum anomaly detection (QAD) can be useful. First, in combination with quantum machine learning algorithms, QAD can question the quality of the decision, just as a safety net to prevent overconfident or useless decisions [11]. The second advantage lies in the $\log_2$ amount of qubits to represent the data compared to the original representation. For example, in the experiments, the wine and iris datasets are represented as 12- and 28-dimensional feature vector (details are in the experimental Sec. III C), whereas only four and five qubits are needed on a quantum device.

We therefore propose to optimize an NF using quantum gates and use the resulting architectures for anomaly detection. In the experiments, we compare the resulting quantum architectures with standard approaches for anomaly detection, e.g., based on isolation forests (IF), local outlier factors (LOF), and one-class support vector machines (SVMs) and show a competitive performance. These methods have been selected since previous works already presented quantum implementations for these variants, or one is in general possible as summarized in Sec. II C 1. We additionally demonstrate how to use the quantum normalizing flow as a generative model by sampling from the target distribution and evaluating the backward flow. A very recent work in this direction was presented in [12]. For the optimization of the quantum gate order, we rely on a quantum architecture search and directly optimize the gate selection and order on a loss function. In our experiments we will use as loss the Kullback- Leibler (KL) divergence and the cosine dissimilarity.

Our contributions can be summarized as follows.

(1) We propose quantum normalizing flows to compute a bijective mapping from data samples to a normal distribution.

(2) Our optimized models are used for anomaly detection and are evaluated and compared to quantum usable reference methods demonstrating a competitive performance.

(3) Our optimized models are used as a generative model by sampling from the target distribution and evaluating the backward flow.

(4) Our source code for optimization will be made publicly available [13].

II. PRELIMINARIES

In this section we give a brief overview of the quantum framework we use later, a summary on normalizing flows, and provide an overview of existing classical and quantum driven anomaly detection frameworks. Three classical and quantum formalizable methods are later used for a direct comparison with our proposed quantum-flow algorithm, namely, isolation forests, LOFs, and single-class SVMs.

A. Quantum gates and circuits

We focus on the setting where our quantum information processing device is comprised of a set of N *logical* qubits, arranged as a quantum register (see, e.g., [14] for further details). Thus we use a Hilbert space of our system $\mathcal{H} \equiv (\mathbb{C}^2)^{\otimes N} \cong \mathbb{C}^{2^N}$ as algebraic embedding. Therefore, a quantum state vector of a five-qubit register is a unit vector in $\mathbb{C}^{2^5} = \mathbb{C}^{32}$. We further assume that the system is not subject to decoherence or other external noise.

Quantum gates are the basic building blocks of quantum circuits, similar to logic gates in digital circuits [15]. According to the axioms of quantum mechanics, quantum logic gates are represented by unitary matrices so that a gate acting on N qubits is represented by a $2^N \times 2^N$ unitary matrix, a quantum gate sequence comprised of a set of such gates which in return are evaluated as a series of matrix multiplications. A quantum circuit of length L is, therefore, described by an ordered tuple $[O(1), O(2), \ldots, O(L)]$ of quantum gates; the resulting unitary operation U implemented by the circuit is the product

$$U = O(L)O(L-1)\cdots O(1). \tag{1}$$

Standard quantum gates include the Pauli-(X, Y, Z) operations, as well as Hadamard, CNOT, SWAP, phase-shift, and TOFFOLI gates, all of which are expressible as standardized unitary matrices [16]. The action of a quantum gate is extended to a register of any size exploiting the tensor product operation in the standard way. Even though some gates do not involve additional variables, however, a phase-shift gate $R_X(\theta)$ applies a complex rotation and involves the rotation angle θ as free parameter.

B. Normalizing flows

A NF is a transformation of a provided (simple) probability distribution (e.g., a normal distribution) into an arbitrary distribution by a sequence of invertible mappings. They were introduced by Rezende and Mohamed [17] as a generative model to generate examples from sampling a normal distribution. Compared to other generative models such as variational autoencoders (VAEs) [18] or generative adversarial networks (GANs) [19] an NF comes along with the property that it maps bijectively and is bidirectionally executable. Thus, the input dimension is the same as the output dimension. Usually, they are optimized via maximum likelihood training and the minimization of a KL-divergence measure. Given two probability distributions P and Q, the Kullback-Leibler divergence (KL divergence) or relative entropy is a dissimilarity measure for two distributions

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x\in\mathcal{X}} p(x)\ln\left(\frac{p(x)}{q(x)}\right). \tag{2}$$

Since the measure is differentiable, it has been frequently used in the context of neural network models for a variety of downstream tasks such as image generation, noise modeling, video generation, audio generation, graph generation, and more [10]. Other dissimilarity measures can be based on the cosine-divergence, optimal transport or the χ^2 measure. Specifically, the cosine-divergence is a useful alternative for quantum computers, as this measure is also directly expressible as quantum gate sequence [20]. The cosine dissimilarity of two unit vectors can be expressed as its simple scalar product (denoted as $\cdot$)

$$D_{\cos}(P, Q) = 1 - P \cdot Q. \tag{3}$$

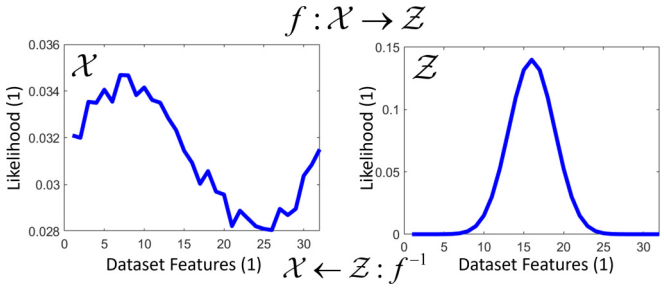

FIG. 1. Visualization of a normalizing flow f. It is a bijective transformation f of an arbitrary distribution (from a given dataset) to a normal distribution.

After the training process, a learned NF can be used in two ways, either as generator or likelihood estimator. The forward pass $f : \mathcal{X} \to \mathcal{Z}$ allows for computing the likelihood of observed data points given the target distribution p_Z (e.g., unit Gaussian). The backward pass $f^{-1} : \mathcal{Z} \to \mathcal{X}$ allows for generating new samples in the original space $\mathcal{X}$ by sampling from the latent space $\mathcal{Z}$ according to the estimated densities as in [21,22]. Figure 1 visualizes the change of variables given the source distribution $p_{\mathcal{X}}$ and the target distribution $p_{\mathcal{Z}}$.

Several neural network architectures for such transformations were proposed in the past [23–26]. Note that all of these architectures allow for learning highly nonlinear mappings, a property which is by definition not possible when solely quantum gates are used (which all comprise linear mappings). We therefore perform later on experiments solely on methods where quantum solutions are presented. Still, since quantum gates are by definition expressible as unitary matrices which are invertible, a quantum gate fulfills the basic properties of an invertible and bijective mapping. Thus, we aim for optimizing a quantum architecture providing the desired NF transformation and use the likelihood estimation from the forward pass to compute an anomaly score for anomaly detection.

C. Anomaly detection

Anomaly detection is the task to identify data points, entities, or events that fall outside an expected range. Anomaly detection is applicable in many domains and can be seen as a subarea of unsupervised machine learning. In our setting we focus on a setting where only positive examples are provided which is also termed one-class classification or novelty detection. In the following we will first summarize recent quantum approaches for anomaly detection and will then introduce in more detail three reference methods we will use in our experiments.

1. Quantum anomaly detection

One of the first approaches to formulate anomaly detection on quantum computers was proposed in [27]. The authors

mainly built up on a kernel principal component analysis and a one-class support vector machine, one of the approaches we will introduce in more detail later. The so-called change point detection was analyzed in [28–30]. In [31], Liang *et al.* proposed anomaly detection using density estimation. Therefore it was assumed that the data followed a specific type of distribution, in its simplest form a Gaussian mixture model. In [32] the LOF algorithm [33] was used and remapped to a quantum formulation. As explained later, the LOF algorithm contains three steps, (i) determine the k-distance neighborhood for each data point x, (ii) compute the local reachability density of x, and (iii) calculate the local outlier factor of x to judge whether x is abnormal. In [34] the authors proposed an efficient quantum anomaly detection algorithm based on density estimation which was driven from amplitude estimation. They showed that their algorithm achieved exponential speed up on the number of training data points M over its classical counterpart. In addition to the fundamental theoretical concepts, many works do not show any experiments on real datasets and are therefore often limited to very simple and artificial examples. Also the generated quantum gate sequences can require a large amount of qubits and they lead to reasonable large code lengths which is suboptimal for real-world scenarios [27,32].

In the following we will summarize three classical and well-established methods which we will later use for a direct comparison to our proposed quantum flow. For the experiments we use the implementation of these algorithms provided by MATLAB [35]. The optimization on the used datasets is very fast and takes less than 1 second on a standard notebook.

2. Isolation forests

An isolation forest is an algorithm for anomaly detection which has been initially proposed by Liu *et al.* [36]. It detects anomalies using characteristics of anomalies, i.e., being few and different. The idea behind the isolation forest algorithm is that anomalous data points are easier to separate from the rest of the data. To isolate a data point, the algorithm generates partitions on the samples by randomly selecting an attribute and then randomly selecting a split value in a valid parameter range. The recursive partitioning leads to a tree structure and the required number of partitions to isolate a point corresponds to the length of the path in the tree. Repeating this strategy leads to an isolation forest, and finally, all path lengths in the forest are used to determine an anomaly score. The isolation forest algorithm computes the anomaly score $s(x)$ of an observation x by normalizing the path length $h(x)$:

$$s(x) = 2^{-\frac{E[h(x)]}{c(n)}}, \tag{4}$$

where $E[h(x)]$ is the average path length over all isolation trees in the isolation forest and $c(n)$ is the average path length of unsuccessful searches in a binary search tree of n observations.

The algorithm was extended in [37] and [38] to address clustered and high-dimensional data. Another extension is anomaly detection for dynamic data streams using random cut forests, which was presented in [39]. In [40] quantum decision trees ereween proposed, which are the basis for an optional quantum isolation forest.

3. Local outlier factor

The LOF algorithm was introduced in [33]. Outlier detection is based on the relative density of a data point with respect to the surrounding neighborhood. It uses the k-nearest neighbor and can be summarized as

$$\mathrm{LOF}_k(p) = \frac{1}{|N_k(p)|} \sum_{o \in N_k(p)} \frac{d_{lrk}(o)}{d_{lrk}(p)}. \tag{5}$$

Here, d_{lrk} denotes the local reachability density and $N_k(p)$ represents the k-nearest neighbor of an observation p. The reachability distance of observation p with respect to observation o is defined as

$$\tilde{d}_k(p, o) = \max[d_k(o), d(p, o)], \tag{6}$$

where $d_k(o)$ is the kth smallest distance among the distances from the observation o to its neighbors, and $d(p, o)$ denotes the distance between observation p and observation o. The local reachability density of observation p is reciprocal to the average reachability distance from observation p to its neighbors

$$d_{lrk}(p) = \frac{1}{\frac{\sum_{o \in N_k(p)} \tilde{d}_k(p,o)}{|N_k(p)|}}. \tag{7}$$

The LOF can be computed on different distance metrics, e.g., an Euclidean, mahalanobis, city block, Minkowsky distance, or others. In [32] a quantum LOF algorithm was presented.

4. Single class SVM

SVMs for novelty detection were proposed in [7]. The idea was to estimate a function f which is positive on a simple set S and negative on the complement, thus the probability that a test point drawn from a probability distribution P lies outside of S equals some *a priori* specified v between 0 and 1. Let $x_i \in \mathbb{R}^N$ denote the training data and Φ be a feature map into a dot product space F such that a kernel expression

$$k(x, y) = \Phi(x)^T \Phi(y) \tag{8}$$

can be used to express a nonlinear decision plane in a linear fashion, which is also known as *kernel trick* [41]. The following formulation optimizes the parameters w and ρ and returns a function f that takes the value $+1$ in a *small* region capturing most of the data points and -1 elsewhere. The objective function can be expressed as quadratic program of the form

$$\min_{w \in F, \xi \in \mathbb{R}^l, \rho \in \mathbb{R}} \quad \frac{1}{2}\|w\|^2 + \frac{1}{v^l} \sum_i \xi_i - \rho \tag{9}$$

$$\text{s.t.} \quad (w^T \Phi(x_i)) \geqslant \rho - \xi_i, \quad \xi \geqslant 0. \tag{10}$$

The nonzero slack variables ξ_i act as penalizer in the objective function. If w and ρ can explain the data, the decision function

$$f(x) = \mathrm{sgn}((w^T \Phi(x)) - \rho) \tag{11}$$

will be positive for most examples x_i, while the support vectors in $\|w\|$ will be minimized, thus the tradeoff between an optimal encapsulation of the training data and accepting

outlier in the data is controlled by v. Expanding f by using the dual problem leads to

$$f(x) = \mathrm{sgn}((w^T \Phi(x)) - \rho) \tag{12}$$

$$= \mathrm{sgn}\left(\sum_i \alpha_i k(x_i, x) - \rho\right), \tag{13}$$

which indicates the support vectors of the decision boundary with nonzero α_is. This basic formulation was frequently used in anomaly detection and established as a well performing algorithm [9,42]. For more details in linear and quadratic programming we refer to [43–46]. Quantum SVMs have been matter of research over several years [47–49]. An approach for a quantum one-class SVM was presented in [27].

III. METHOD

In the following we present the proposed method and conducted experiments. The section starts with a brief summary of the used optimization strategy based on quantum architecture search, continues with the evaluation metrics, based on a so-called receiver operating characteristic curve (ROC), and presents the proposed quantum flow anomaly detection framework. The evaluation is performed on two selected datasets, the iris dataset and the wine dataset. Here, we will compare the outcome of our optimized quantum flow with the performance of isolation forests, the LOF and a single class SVM.

A. Optimization

Optimization is based on quantum architecture search. The name is borrowed and adapted from *Neural Architecture Search* (NAS) [50,51], which is devoted to the study and hyperparameter tuning of neural networks. Many QAS variants are focused on discrete optimization and exploit optimization strategies for nondifferentiable optimization criteria. In the past, variants of Gibbs sampling [52], evolutional approaches [53], genetic algorithms [54,55], and neural-network based predictors [56] were suggested. A recent survey on QAS can be found in [57]. For this work, we rely on the previous work [58] proposing Monte Carlo graph search. The optimized loss function is in our case the Kullback-Leibler divergence, $D_{\mathrm{KL}}(P \parallel Q)$ [see Eq. (2)], which is the standard loss for many anomaly detection frameworks. We will also perform experiments using the cosine-similarity measure. As discussed later, this measure can be evaluated directly from quantum states via a SWAP test. Thus, our approach can be executed on a quantum computer with a sequence comprising of state preparation, the optimized quantum gates sequences and followed by evaluating the quantum cosine dissimilarity as proposed in [20]. A quantum architecture search requires at the end (i) a pool of elementary quantum gates $\mathcal{OP} = \{O_1, O_2, \ldots\}$ to sample from, (ii) a loss function, and (iii) some simple hyperparameters, e.g., the maximum length of the quantum gate sequence or a stopping criteria. Then different gate orders are sampled and evaluated. The used quantum architecture search algorithm is based on Monte Carlo graph search (MCGS) [59,60] and measures of importance sampling. MCGS is very efficient, since many operators for quantum computing act locally (e.g., the H gates, X gates, Z gates, and more). Therefore, many combinations of sampled unitary matrices representing the gate order are algebraically commutative. The MCGS-algorithms can be motivated from Monte Carlo tree search (MCTS). It is a heuristic search algorithm for decision processes [60]. It makes use of random sampling and balances the exploration-exploitation dilemma in large search spaces. MCTS can be very efficient as it visits more interesting subtrees more often. Thus, it grows asymmetrically and focuses the search time on more relevant parts of the tree. For the MCGS, we start with the identity operator $\mathbb{I}$ and build quantum circuits by selecting elementary gates from a predefined set $\mathcal{OP} = \{O_1, O_2, \ldots\}$ of elementary quantum gates that we are allowed to apply. Due to the universality theorem [61] it is possible to approximate any unitary matrix to arbitrarily good accuracy by using a sufficiently long product of such gates. The general idea is to grow a graphical model with nodes containing unitary matrices and edges encoding a unitary operator $O_i \in \mathcal{OP}$. The graph is initialized with the identity matrix $\mathbb{I}$ as root node. An operator O_i is selected and applied to the root node. This yields a new node by multiplying the selected operator with the unitary matrix of the parent node (which is the identity matrix in the beginning). If the resulting unitary is already existing as node in the graph, a direct edge from the parent to the already existing node can be added. In addition, a new node is generated and connected with the parent node. Thus, while growing the graph, the resulting unitary matrices are provided as graph nodes and the underlying quantum code can be computed by finding the shortest path from the root node to the target unitary and by collecting the operators along the edges of the path. Figure 2 is taken from [58] and shows the general principle. Thus, each node is identified with a possible quantum circuit. Please note, that this graph contains cycles since identical quantum circuits have multiple representations with different gates and gate orders. The remaining challenge is to grow the graph in an efficient manner. Given a specific task, every node will receive a quality score which is used to compute a probability for its selection and development along this graph area. Poisson sampling is then exploited as the underlying sampling process for selecting a vertex to further develop. It is the basic paradigm of Monte Carlo search [62] and adapted Gibbs sampling [63] to iteratively grow a graph. We refer to [58] for further details.

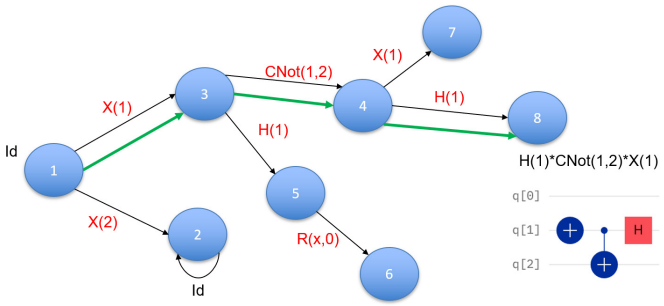

FIG. 2. Tiny example graph of quantum circuits. The edges are labeled with elementary gates. The vertices are given by the unitary operator built by taking the product of the gates along the shortest path. (Image taken from [58].)

B. Evaluation metrics

To compare the performance of different algorithms, the area under the ROC curve (receiver operating characteristic curve) [64] is a common measure. To summarize, a ROC curve is a graph showing the performance of a classification model at all classification thresholds. In our case, the classification threshold is the anomaly score of the algorithm and the x-y axes contains the false-positive rate (x axis) versus the true-positive rate (y axis) for all possible anomaly thresholds. The area under the generated curve is the AUROC which is 1 in the optimal case when all examples are correctly classified while producing no false positives. For anomaly detection, the AUROC is the standard measure in the literature for the following reasons, (i) the AUROC is scale invariant. It measures how well predictions are ranked, rather than their absolute values and (ii) the AUROC is classification-threshold invariant. It measures the quality of the model's predictions irrespective of what classification threshold is chosen.

C. Quantum flow anomaly detection

For the experiments, the classical *wine* and *iris* datasets were used. The datasets present multicriterial classification tasks, with three categories for the wine dataset, and three for the iris dataset. The datasets are all available at the UCI repository [65]. To model an anomaly detection task using a quantum circuit, first the data is encoded as a higher-dimensional binary vector. Taking the iris dataset as a toy example, it consists of four-dimensional data encoding *sepal length*, *sepal width*, *petal length,* and *petal width*. A kMeans clustering on each dimension with $k = 3$ is used on the training data. Thus, every datapoint can be encoded in a $4 \times 3 = 12$-dimensional binary vector which contains exactly four nonzero entries. The iris dataset contains three categories, namely *setosa, versicolor,* and *virginica*. We select 50% of data points from one class (e.g., *setosa*) for training and use the remaining datapoints, as well as a second class (e.g., *virginica*) for testing. Thus, the *setosa* test cases should be true positives, whereas the *virginica* should be correctly labeled as anomalies. Figure 3 shows a TSNE-plot (t-distributed stochastic neighbor embedding plot) [66] of the iris dataset on the top and a TSNE plot of the wine dataset on the bottom. The iris dataset was selected since one class separates very easily from the rest, whereas the remaining classes are more similar and overlapping. The second class of the wine dataset is spreading into the classes one and three, here the anomaly detection is also challenging. These properties will also be reflected in the anomaly scores in the experiments.

For the classically computed quantum architecture search, we compute the discrete distribution N_{hist} as the normalized histogram of the training dataset as input and use a binomial distribution ($p = 0.5$) as target

$$X \sim p(X == k) \sim N_{\text{hist}}(k, n), \tag{14}$$

$$Y \sim p(Y == k) \sim \binom{n}{k} p^k (1-p)^{n-k}. \tag{15}$$

The optimization task is to find an ordered set of quantum gates [see Eq. (1)] which lead a unitary matrix $U = O(L)$ $O(L\text{-}1)\cdots O(1)$ which minimizes the KL divergence of the transformed input distribution.

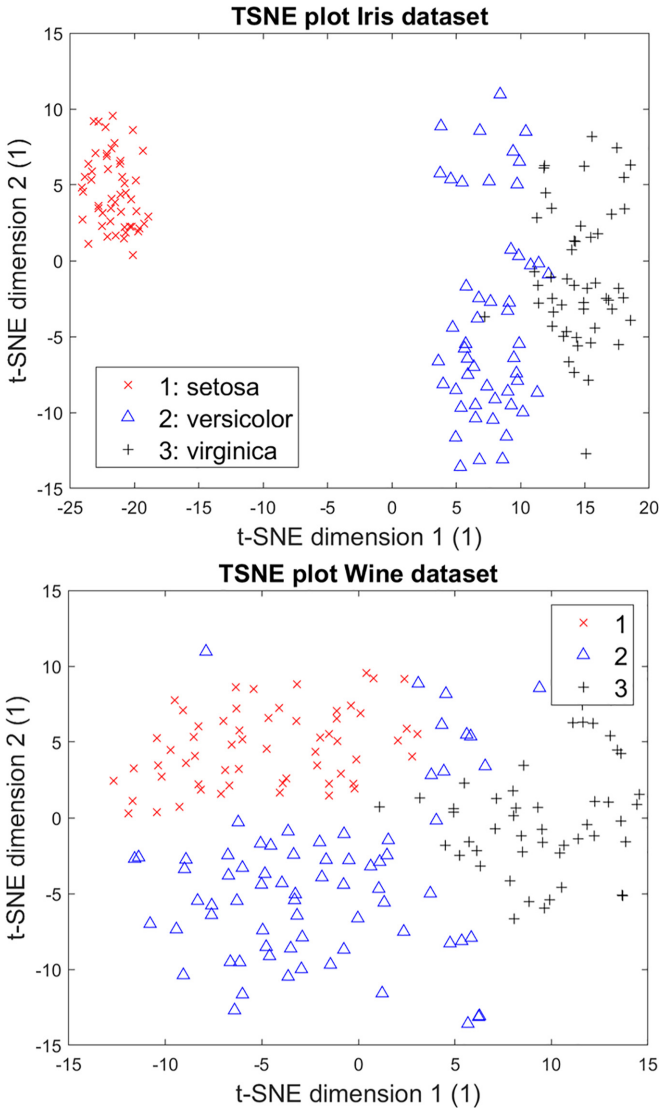

FIG. 3. TSNE-plot of the iris (top) and wine (bottom) datasets. The iris dataset has been selected since one class separates very easy from the rest, whereas the remaining classes are more similar and overlapping. The second class of the wine dataset is spreading into the classes one and three.

$$\min_{U=O(L)O(L-1)\cdots O(1)} D_{\text{KL}}((|UX|) \parallel Y). \tag{16}$$

Figure 4 shows an example of how an input distribution is transformed towards a target distribution using an optimized quantum gate sequence U. The KL divergence between both distributions is used as anomaly score. Here the ROC curve evaluates all optional thresholds used for anomaly detection as a scale-invariant measure. This is useful, as it measures how well predictions are ranked, rather than their absolute values.

To make the resulting code more suitable for a quantum implementation, we also optimized the cosine-dissimilarity

$$\min_{U=O(L)O(L-1)\cdots O(1)} D_{\cos}((|UX|), Y), \tag{17}$$

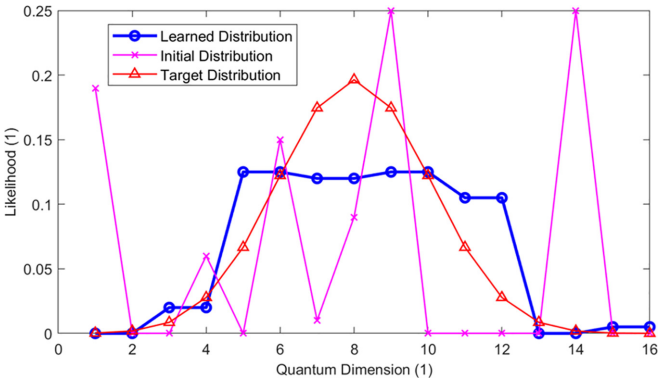

FIG. 4. Optimized NF from an input distribution (magenta cross) to a target distribution (blue circle), and the comparison to a discrete Gaussian distribution (red triangle).

and evaluate the performance. It turns out that the obtained results for using a KL divergence and cosine score only deviate up to some noise.

The quantum architecture search algorithm requires as input a pool of optional quantum gates to select from. In our experiments we used the Pauli-(X, Y, Z) operations, as well as Hadamard, CNOT, SWAP, phase-shift, and TOFFOLI gates. The phase shift gates have a continuous parameter θ we sampled with $\pi, \frac{\pi}{2}, \frac{\pi}{4}$. Thus, only the discrete ordering and selection of quantum gates is optimized by the quantum architecture search. Table I summarizes the used datasets, the settings and the train and test splits. Across all experiments, the optimization of the quantum circuits requires between 10 seconds and 1 minute. Please note that this is only the (offline) training stage. Inference on a quantum computer does not require further optimization. Figure 5 summarizes the resulting ROC curves and the area under the ROC curve as a final quality measure. Additionally, we provide the ROC curves and the obtained results of isolation forests, the LOF and the single-class SVM. The final performance is also summarized in Table II. It is apparent that our proposed optimized quantum flows can achieve competitive performance. Additionally, the code is available as short and highly performant quantum gate sequence. The obtained quantum gate sequences are provided in Fig. 6. In the following section we will discuss how the decision algorithm can be implemented on a quantum computer.

D. Quantum implementation

The anomaly detection algorithm described in this work essentially consists of two parts. First, we use quantum architecture search to find the unitary that models the normalizing flow. In the second step we evaluate the normalizing flow on new samples to detect a potential anomaly. In this section we discuss how this second part can also efficiently handled by a quantum computer. First, starting from the sample we need to prepare the input to our unitary. A common assumption, see e.g., [27], Sec. III], is that we have access to the unitary that prepares the input state $|\Phi\rangle = V|0\rangle$. Note, however, that typical classical datasets, such as the examples discussed in this work, often give sparse vectors that can, also without the previous assumption, be encoded efficiently into a pure quantum state [67].

Next, we implement the normalizing flow unitary as for the case of our examples given in Fig. 6. These consist of elementary gates and the circuit depth is a parameter that can be set in the quantum architecture search. The resulting state is again a pure state, given by $|\psi\rangle = U|\Phi\rangle$.

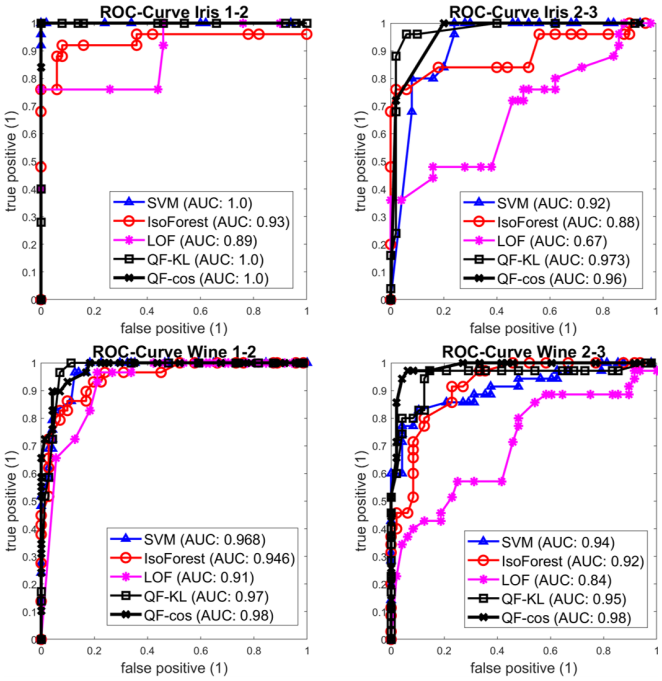

FIG. 5. The resulting ROC curves and the aurea-under the ROC curve (AUROC) for different settings on the Iris and Wine dataset. QF-KL shows the performance using the KL divergence for optimization and QF-cos shows the performance using the cosine dissimilarity score.

TABLE I. Datasets overview, the used normal class, the anomal class, and train or test splits.

Dataset	Dim	BDim	qubits	# Train	# Test
Iris 1–2	4	12	4	25	25/50
Iris 2–3	4	12	4	25	25/50
Wine 1–2	14	28	5	29	29/71
Wine 2–3	14	28	5	35	35/48

TABLE II. Area under ROC performance on different anomaly detection cases for isolation forests, single-class SVMs, and our proposed quantum normalizing flow. QF-KL shows the performance using the KL divergence for optimization and QF-cos shows the performance using the cosine dissimilarity score. Highest score per dataset highlighted in bold.

Dataset	iso-Forest	LOF	SVM	QF-KL	QF-cos
Iris 1–2	0.93	0.89	**1.0**	**1.0**	**1.0**
Iris 2–3	0.88	0.67	0.92	**0.97**	0.96
Wine 1–2	0.946	0.91	0.968	0.97	**0.979**
Wine 2–3	0.92	0.84	0.94	0.95	**0.98**

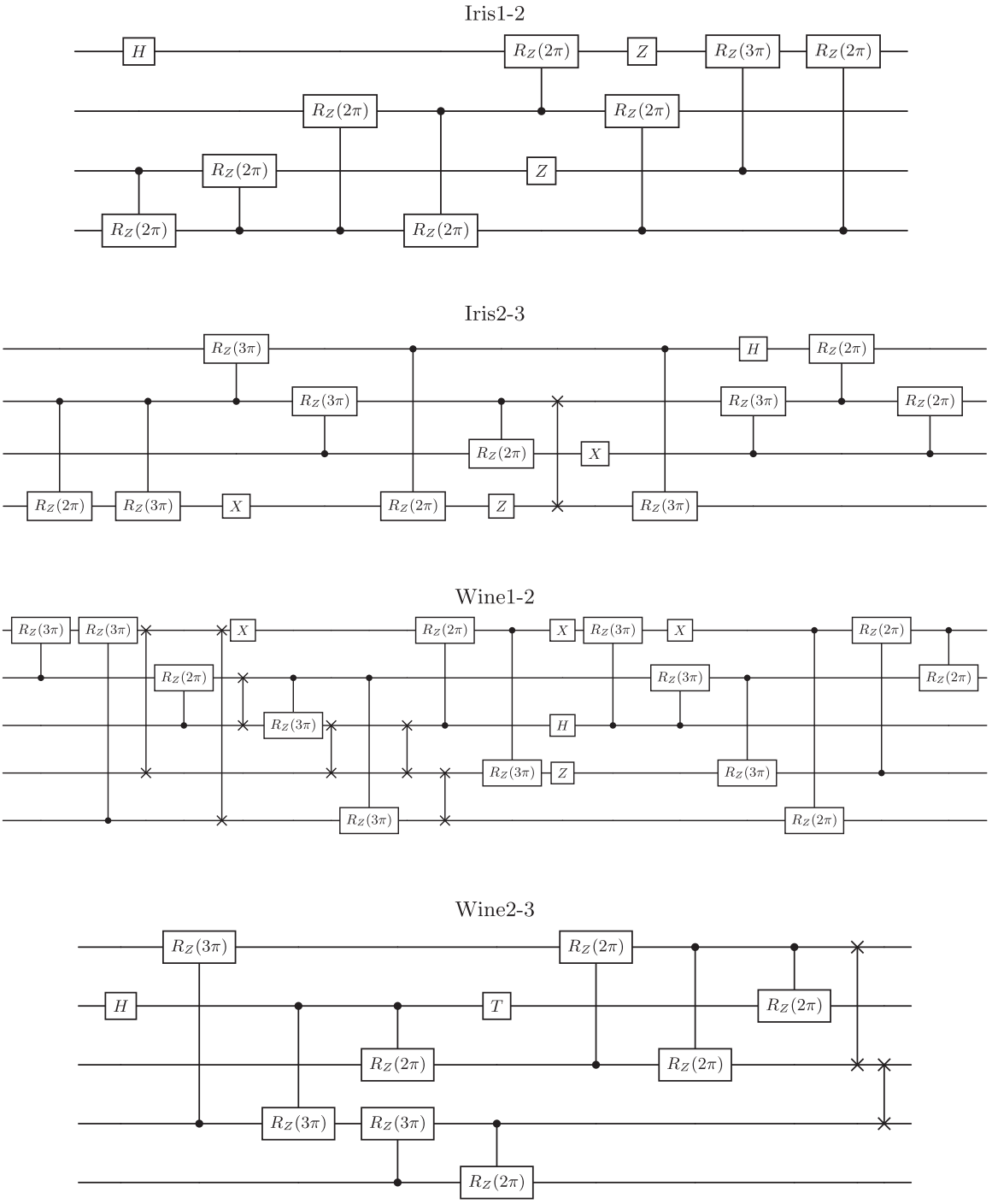

FIG. 6. Obtained quantum gate sequences after optimization. Despite the competitive performance the resulting codes are reasonable small and efficient.

The final step is to determine the similarity with the target normal distribution. In general, reading out the exact state of $|\psi\rangle$ can be challenging, e.g., using tomography would require an exponential number of measurements. We hence focus on the cosine-dissimilarity measure. By its definition, it sufficient to compute overlap of $|\psi\rangle$ with the target distribution. To avoid reading out the $|\psi\rangle$, we can instead preprare a state $|\phi\rangle$ according to the normal distribution in Eq. (15) and then apply a simple SWAP test [68].

This requires one auxilliary qubit, two Hadamard gates, a controlled SWAP operation, and a final quibit measurement. The implementation is shown in Fig. 7. The result of the measurement is a binary random variable with probability

$$p = \tfrac{1}{2} - \tfrac{1}{2}|\langle\psi|\phi\rangle|^2, \tag{18}$$

allowing us to approximate the scalar product $|\langle\psi|\phi\rangle|^2$ up to error ϵ with $O(\frac{1}{\epsilon^2})$ samples. In summary, we have an algorithm that after initial classical optimization can efficiently run on

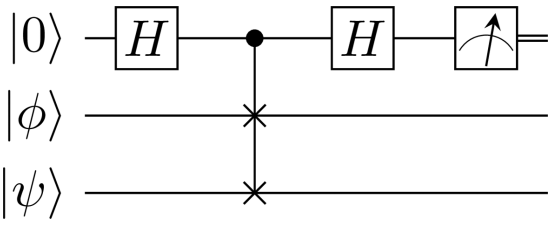

FIG. 7. Implementation of the SWAP test for two quantum states ψ and ϕ.

a quantum computer and reaches higher performance then previous algorithms with known quantum implementations.

E. Normalizing flow as a generator

As already mentioned in Sec. II.2, a normalizing flow can be used for several purposes, e.g., anomaly detection as shown in the previous part. Another application is to use the normalizing flow as a generator by sampling from the normal distribution and inverting the forward transformation. Thus, in the final experiment, a normalizing flow has been trained on the first class of both, the iris and wine dataset, respectively. Afterwards, samples from the normal distribution are generated and transformed using the inverse flow which leads to new samples in the original data. Figure 8 visualizes the TSNE plots for these datasets (in crosses) as well as generated examples (in circles) after sampling complex values from the normal distribution and inverting the learned forward mapping U. Note, that the backward transformation can lead to nonuseful samples since only the distribution on the absolute values is optimized in the forward flow. Thus, sampling complex numbers can lead to unlikely examples. We therefore verify the samples by backprojecting them onto the normal distribution again and only select samples which have a small KL divergence. As expected, the generated samples are in the domain of the training data and generalize examples among them. Figure 8 shows the TSNE data for the three classes and in circles the generated samples which are located around the first class label.

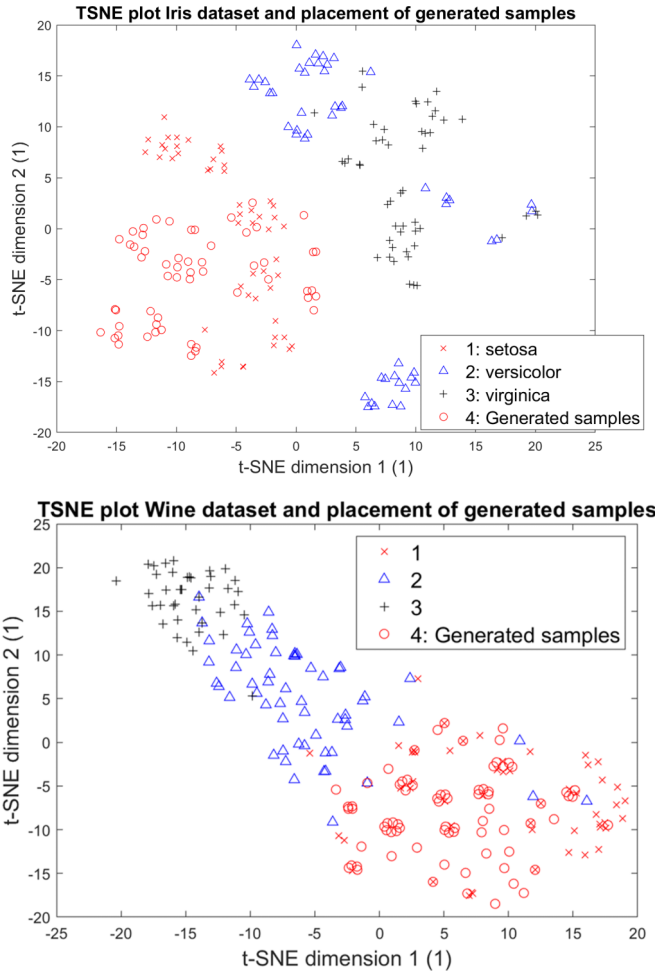

FIG. 8. A NF has been trained on the first class for the iris and wine dataset, respectively. The crosses denote the TSNE plot of the training data. The generated examples from sampling the normal distribution are shown in circles and consistently fall into the learned domain.

IV. SUMMARY

In this work, quantum architecture search is used to compute a normalizing flow which can be summarized as a bijective mapping from an arbitrary distribution to a normal distribution. The optimization is based on a Kullback-Leibler divergence or the cosine dissimilarity. Once such a mapping has been optimized, it can be applied to anomaly detection by comparing the distribution of quantum measurements to the expected normal distribution. In the experiments we perform comparisons to three standard methods, namely, isolation forests, the LOF and a single-class SVM. The optimized architectures show competitive performance despite being fully implementable on a quantum computer. Additionally we demonstrate how to use the normalizing flow as generator by sampling from a normal distribution and inverting the flow.

ACKNOWLEDGMENTS

This work was supported, in part, by the Quantum Valley Lower Saxony and under Germany's Excellence Strategy EXC-2122 PhoenixD. We would like to thank Tobias Osborne (Institute for Theoretical Physics at Leibniz University Hannover) for the fruitful discussions and advice on preparing the manuscript.

[1] M. Evangelou and N. M. Adams, An anomaly detection framework for cyber-security data, Computers Security **97**, 101941 (2020).

[2] T. Fernando, S. Denman, D. Ahmedt-Aristizabal, S. Sridharan, K. R. Laurens, P. Johnston, and C. Fookes, Neural memory plasticity for medical anomaly detection, Neural Networks **127**, 67 (2020).

[3] J. T. Brockmann, M. Rudolph, B. Rosenhahn, and B. Wandt, *The Voraus-Ad Dataset for Anomaly Detection in Robot Applications, Transactions on Robotics* (IEEE, New York, 2023).

[4] S. Reddy, P. Poduval, A. V. Singh Chauhan, M. Singh, S. Verma, K. Singh, and T. Bhowmik, Tegraf: Temporal and graph based fraudulent transaction detection framework, in *Proceedings of the Second ACM International Conference on AI in Finance, ICAIF '21* (Association for Computing Machinery, New York, 2022).

[5] M. Rudolph, B. Wandt, and B. Rosenhahn, Same same but differnet: Semi-supervised defect detection with normalizing flows, arXiv:2008.12577.

[6] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft, Deep semi-supervised anomaly detection, arXiv:1906.02694.

[7] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, Support vector method for novelty detection, in *Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS'99* (MIT Press, Cambridge, MA, 1999), pp. 582–588.

[8] C. Scott and G. Blanchard, Novelty detection: Unlabeled data definitely help, in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, edited by D. van Dyk and M. Welling (Proceedings of Machine Learning Research, Hilton Clearwater Beach Resort, Clearwater Beach, Florida, USA, 2009), Vol. 5, pp. 464–471.

[9] M. Amer, M. Goldstein, and S. Abdennadher, Enhancing one-class support vector machines for unsupervised anomaly detection, in *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description, ODD '13* (ACM, New York, 2013), pp. 8–15.

[10] I. Kobyzev, S. J. D. Prince, and M. Brubaker, Normalizing flows: An introduction and review of current methods, IEEE Transactions on Pattern Analysis and Machine Intelligence **43**, 3964 (2021).

[11] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. Rajendra Acharya, V. Makarenkov, and S. Nahavandi, A review of uncertainty quantification in deep learning: Techniques, applications and challenges, Information Fusion **76**, 243 (2021).

[12] A. Rousselot and M. Spannowsky, Generative invertible quantum neural networks, SciPost Phys. **16**, 146 (2024).

[13] http://www.TBA(after-acceptance).

[14] P. Kaye, R. Laflamme, and M. Mosca, *An Introduction to Quantum Computing* (Oxford University Press, New York, 2007).

[15] P. Selinger, Towards a quantum programming language, Mathematical Structures in Computer Science **14**, 527 (2004).

[16] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2010).

[17] D. Rezende and S. Mohamed, Variational inference with normalizing flows, in *International Conference on Machine Learning* (PMLR, 2015), pp. 1530–1538.

[18] D. P. Kingma and M. Welling, Auto-encoding variational bayes, arXiv:1312.6114.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative Adversarial Nets, in Advances in Neural Information Processing Systems* (2014), pp. 2672–2680.

[20] D. Pastorello and E. Blanzieri, A quantum binary classifier based on cosine similarity, in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, New York, 2021), pp. 477–478.

[21] D. P. Kingma and P. Dhariwal, Glow: Generative flow with invertible 1x1 convolutions, arXiv:1807.03039.

[22] T. Wehrbein, M. Rudolph, B. Rosenhahn, and B. Wandt, Probabilistic monocular 3D human pose estimation with normalizing flows, in *Proceedings of the IEEE International Conference on Computer Vision* (IEEE, New York, 2021).

[23] L. Dinh, J. Sohl-Dickstein, and S. Bengio, Density estimation using real-NVP, arXiv:1605.08803.

[24] M. Germain, K. Gregor, I. Murray, and H. Larochelle, Made: Masked autoencoder for distribution estimation, in *International Conference on Machine Learning* (2015), pp. 881–889.

[25] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, Improved variational inference with inverse autoregressive flow, in *Advances in Neural Information Processing Systems* (ACM, New York, 2016), pp. 4743–4751.

[26] G. Papamakarios, T. Pavlakou, and I. Murray, Masked autoregressive flow for density estimation, arXiv:1705.07057.

[27] N. Liu and P. Rebentrost, Quantum machine learning for quantum anomaly detection, Phys. Rev. A **97**, 042315 (2018).

[28] M. Fanizza, C. Hirche, and J. Calsamiglia, Ultimate limits for quickest quantum change-point detection, Phys. Rev. Lett. **131**, 020602 (2023).

[29] G. Sentís, J. Calsamiglia, and R. Muñoz Tapia, Exact identification of a quantum change point, Phys. Rev. Lett. **119**, 140506 (2017).

[30] G. Sentís, E. Martínez-Vargas, and R. Muñoz Tapia, Online strategies for exactly identifying a quantum change point, Phys. Rev. A **98**, 052305 (2018).

[31] J.-M. Liang, S.-Q. Shen, M. Li, and L. Li, Quantum anomaly detection with density estimation and multivariate Gaussian distribution, Phys. Rev. A **99**, 052310 (2019).

[32] M. Guo, S. Pan, W. Li, F. Gao, S. Qin, X. Yu, X. Zhang, and Q. Wen, Quantum algorithm for unsupervised anomaly detection, Physica A **625**, 129018 (2023).

[33] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, LOF: Identifying density-based local outliers, SIGMOD Rec. **29**, 93 (2000).

[34] M. Guo, H. Liu, Y. Li, W. Li, F. Gao, S. Qin, and Q. Wen, Quantum algorithms for anomaly detection using amplitude estimation, Physica A **604**, 127936 (2022).

[35] The MathWorks Inc, MATLAB version: 23.2.0.2409890, https://in.mathworks.com/matlabcentral/answers/2046010-mac-and-linux-users-please-support-me-with-a-very-simple-task-videowriter-getprofiles (2023).

[36] F. T. Liu, K. M. Ting, and Z.-H. Zhou, Isolation forest, in *2008 Eighth IEEE International Conference on Data Mining* (IEEE, New York, 2008), pp. 413–422.

[37] F. T. Liu, K. Ting, and Z.-H. Zhou, *On Detecting Clustered Anomalies Using Sciforest* (Springer, New York, 2010), Vol. 6322, pp. 274–290, 09.

[38] R. J. Hyndman, P. D. Talagala and K. Smith-Miles, Anomaly detection in high-dimensional data, Journal of Computational and Graphical Statistics **30**, 360 (2021).

[39] S. Guha, N. Mishra, G. Roy, and O. Schrijvers, Robust random cut forest based anomaly detection on streams, in *Proceedings of The 33rd International Conference on Machine Learning*, edited by M. F. Balcan and K. Q. Weinberger (Proceedings of Machine Learning Research, New York, USA, 2016), Vol. 48, pp. 2712–2721.

[40] S. Lu and S. L. Braunstein, Quantum decision tree classifier, Quant. Inf. Process. **13**, 757 (2014).
[41] M. A. Aizerman, E. A. Braverman, and L. Rozonoer, Theoretical foundations of the potential function method in pattern recognition learning, in *Automation and Remote Control*, number 25 in Automation and Remote Control (Springer, New York, 1964), pp. 821–837.
[42] A. Rahimi and B. Recht, Random features for large-scale kernel machines, in *Advances in Neural Information Processing Systems*, edited by J. Platt, D. Koller, Y. Singer, and S. Roweis (Curran Associates, Red Hook, NY, 2007), Vol. 20.
[43] K. G. Murty, *Linear Programming* (Wiley, New York, 1983).
[44] H. T. Nguyen and K. Franke, A general lp-norm support vector machine via mixed 0-1 programming, in *Machine Learning and Data Mining in Pattern Recognition* (Springer, Berlin, 2012), pp. 40–49.
[45] W. H. Paul, *Logic and Integer Programming*, 1st ed. (Springer, New York, 2009).
[46] B. Rosenhahn, Optimization of sparsity-constrained neural networks as a mixed integer linear program, J. Optim. Theory Appl. **199**, 931 (2023).
[47] C. Ding, T.-Y. Bao, and H.-L. Huang, Quantum-inspired support vector machine, IEEE Transactions on Neural Networks and Learning Systems **33**, 7210 (2022).
[48] J. Heredge, C. D. Hill, L. C. L. Hollenberg, and M. Sevior, Quantum support vector machines for continuum suppression in b meson decays, *Computing and Software for Big Science* (Springer, New York, 2021), Vol. 5.
[49] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big data classification, Phys. Rev. Lett. **113**, 130503 (2014).
[50] R. Miikkulainen, *Neuroevolution* (Springer, New York, 2020), pp. 1–8.
[51] S. Xie, H. Zheng, C. Liu, and L. Lin, SNAS: stochastic neural architecture search, in *International Conference on Learning Representations* (2019).
[52] L. Li, M. Fan, M. Coram, P. Riley, and S. Leichenauer, Quantum optimization with a novel gibbs objective function and ansatz architecture search, Phys. Rev. Res. **2**, 023074 (2020).
[53] L. Franken, B. Georgiev, S. Mucke, M. Wolter, R. Heese, C. Bauckhage, and N. Piatkowski, Quantum circuit evolution on nisq devices, in *2022 IEEE Congress on Evolutionary Computation (CEC)* (IEEE, New York, 2022), pp. 1–8.
[54] U. Las Heras, U. Alvarez-Rodriguez, E. Solano, and M. Sanz, Genetic algorithms for digital quantum simulations, Phys. Rev. Lett. **116**, 230504 (2016).
[55] R. Rasconi and A. Oddi, An innovative genetic algorithm for the quantum circuit compilation problem, Proceedings of the AAAI Conference on Artificial Intelligence **33**, 7707 (2019).
[56] S.-X. Zhang, C.-Y. Hsieh, S. Zhang, and H. Yao, Neural predictor based quantum architecture search, Machine Learning: Science and Technology **2**, 045027 (2021).
[57] W. Zhu, J. Pi, and Q. Peng, A brief survey of quantum architecture search, in *Proceedings of the 6th International Conference on Algorithms, Computing and Systems, ICACS '22* (Association for Computing Machinery, New York, 2023).
[58] B. Rosenhahn and T. J. Osborne, Monte carlo graph search for quantum circuit optimization, Phys. Rev. A **108**, 062615 (2023).
[59] J. Czech, P. Korus, and K. Kersting, Improving alphazero using monte-carlo graph search, in *Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling* 31, 103 (2021).
[60] M. C. Fu, *Monte Carlo Tree Search: A Tutorial, in 2018 Winter Simulation Conference (WSC)* (ACM, New York, 2018), pp. 222–236.
[61] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2000).
[62] N. Metropolis and S. Ulam, The Monte Carlo method, J. Am. Stat. Assoc. **44**, 335 (1949).
[63] E. George and R. E. McCulloch, Variable selection via Gibbs sampling, J. Am. Stat. Assoc. **88**, 881 (1993).
[64] J. A. Hanley and B. J. McNeil, The meaning and use of the area under a receiver operating characteristic (ROC) curve, Radiology **143**, 29 (1982).
[65] D. Dua and C. Graff, UCI machine learning repository, http://archive.ics.uci.edu/ml (2017).
[66] G. E. Hinton and S. Roweis, Stochastic neighbor embedding, in *Advances in Neural Information Processing Systems*, edited by S. Becker, S. Thrun, and K. Obermayer (MIT Press, Cambridge, MA, 2002), Vol. 15.
[67] N. Gleinig and T. Hoefler, An efficient algorithm for sparse quantum state preparation, in *Proceedings of the 2021 58th ACM/IEEE Design Automation Conference (DAC)* (IEEE, New York, 2021), pp. 433–438.
[68] H. Buhrman, R. Cleve, J. Watrous, and R. De Wolf, Quantum fingerprinting, Phys. Rev. Lett. **87**, 167902 (2001).