Formal Verification of Variational Quantum Circuits

NICOLA ASSOLINI*, LUCA MARZARI*, ISABELLA MASTROENI, and ALESSANDRA DI PIERRO, Department of Computer Science, University of Verona, Italy

Variational quantum circuits (VQCs) are a central component of many quantum machine learning algorithms, offering a hybrid quantum-classical framework that, under certain aspects, can be considered similar to classical deep neural networks. A shared aspect is, for instance, their vulnerability to adversarial inputs—small perturbations that can lead to incorrect predictions. While formal verification techniques have been extensively developed for classical models, no comparable framework exists for certifying the robustness of VQCs. Here, we present the first in-depth theoretical and practical study of the formal verification problem for VQCs. Inspired by abstract interpretation methods used in deep learning, we analyze the applicability and limitations of interval-based reachability techniques in the quantum setting. We show that quantum-specific aspects, such as state normalization, introduce inter-variable dependencies that challenge existing approaches. We investigate these issues by introducing a novel semantic framework based on abstract interpretation, where the verification problem for VQCs can be formally defined, and its complexity analyzed. Finally, we demonstrate our approach on standard verification benchmarks.

1 INTRODUCTION

The computational paradigm of Variational Quantum Circuits (VQCs) provides a general framework for solving a wide range of problems by employing the power of quantum computers in synergy with the most advanced classical optimisation tools [14, 25, 34]. VQCs are algorithms inspired by one of the most fundamental principles in Quantum Mechanics, namely the Variational Principle, at the base of any feasible tool for approximating the energy of the ground state of a system. These algorithms are characterized by a modular design, with components that may vary in structure and complexity depending on the specific problem. For their versatility, they have been used to address problems in quantum chemistry, such as calculating molecular energies, in mathematical applications, such as solving systems of equations, in combinatorial optimization, error correction, circuit compilation, etc. VQCs also play a fundamental role in several techniques developed in Quantum Machine Learning [8, 61], for classification tasks and, in general, for learning complex patterns from data. Examples of VQC-based approaches to machine learning include Variational Quantum Support Vector Machines [34] and Quantum Generative Adversarial Networks [23].

Given the widespread use of VQCs in so many application fields, the problem of verifying their properties, such as robustness (also referred to as stability) and accuracy, is crucial. This problem arises from different aspects of the algorithms and their implementations. One aspect is related to the nature of the quantum device used for computation, allowing factors like decoherence and gate errors to introduce errors in the computation and measurement process. These errors propagate throughout the optimization process, impacting the accuracy of the results. Another aspect, which is closer to the specific application employing a VQC, is related to the heuristic nature of these algorithms and the need for an empirical evaluation to assess their properties.

In this paper, we will address the problem of the verification of VQCs relatively to the second aspect mentioned above, with machine learning as a reference application and, in particular, the VQC-based classification method for quantum supervised learning.

To this purpose, a useful source of inspiration is the analogy with a popular method in classical machine learning, the Neural Network model (NN), and in particular with its training process, which

---

*These authors contributed equally to this work.

Authors' address: Nicola Assolini, nicola.assolini@univr.it; Luca Marzari, luca.marzari@univr.it; Isabella Mastroeni, isabella.mastroeni@univr.it; Alessandra Di Pierro, alessandra.dipierro@univr.it, Department of Computer Science, University of Verona, Italy.

is similar to the variational behaviour of a VQC. In fact, at the core of a VQC lies a parameterized quantum circuit with a specific structure, known as *ansatz* or *variational form*, which is formed by a sequence of parameterized gates and provides a trial (starting) point. This is followed by a training procedure that iteratively updates the parameters based on the classical optimization of a problem-specific cost function. This model clearly resembles that of a neural network architecture, in the sense that artificial neural networks, too, are parameterized mathematical models, where parameters are updated during training by means of an optimization procedure. Moreover, VQCs are, like NNs, susceptible to adversarial inputs, i.e., small perturbations to the input state leading to unexpected predictions [65, 70]. Crucially, due to the heuristic and approximate nature of (quantum) training algorithms, the common approach for assessing their robustness is an empirical evaluation, which is obviously insufficient to provide provable guarantees.

Based on this analogy between VQC-based supervised learning and classical neural networks, and inspired by some methods that have been developed for the formal verification of NNs [40], we present a framework for the theoretical and practical analysis of the problem of formally verifying variational quantum circuits.

We focus, in particular, on the investigation of one of the main challenges to the realization of formal verification methods, which, contrary to empirical evaluation, are able to provide provable guarantees, namely the complexity of the verification problem. As for classical NN, we will show in the paper that this problem can be, in general, computationally infeasible when the input space is very large, even if it is discrete and finite, as is the case in most real-world scenarios.

To address this problem, we will adopt the abstract interpretation framework and present a technique for conservatively approximating the behavior of a quantum circuit for the purpose of verifying the accuracy of a VQC-based classifier via the analysis of the approximation error (precision of the abstraction).

At first glance, an abstract approach may appear to be a simpler problem than in the classical case, where the verification of a deep NN has to deal with non-linear activation functions, which typically transform abstract domains (such as polytopes or hyperrectangles) into non-convex shapes that are difficult to analyze. In contrast, the training of a VQC is based on a purely linear learning process based on unitary (and thus convex) operators, represented by unitary matrices, possibly decomposed in a sequence of unitary operators, giving the circuit a layered structure. However, this unitary process can only be applied after transforming the classical input data into quantum states by means of an appropriate encoding phase [58]. If non-linear training does not represent a problem in our setting, we have to deal, nevertheless, with another component of a VQC, which is responsible for transforming classical input data into a quantum state. This transformation, called quantum embedding or quantum feature map, is typically achieved via a (highly) non-linear encoding on the set of inputs [34, 62], and changes the computational arena from the classical dataset to a quantum state space, i.e. a space of complex vectors of norm 1. Normalization introduces strong dependencies among data, which are particularly problematic for the definition of reasonable abstractions.

While in classical deep NNs such dependencies are only a potential issue, in quantum systems this becomes particularly problematic. On top of this issue, another challenge in VQCs is the *measurement* operation performed at the end of the circuit to read the results of the computation. This operation produces a probability distribution over all possible outcomes and is obviously non-invertible: given a probability distribution, there are infinitely many quantum states that could produce it upon measurement. As for non-linear neural networks, this aspect does not allow to perform a backward analysis and determine the sets of inputs that generated the desired output.

All these issues motivate the need for a novel formulation and analysis tailored specifically to VQCs.

Contributions and Structures. After recalling some basic notions of quantum computing, abstract interpretation, and VQC (Section 2 and Section 3), the paper's contributions are as follows. We formally define a **language modeling VQCs** (syntax and semantics), based on a representation of quantum states as functions (Section 4). We introduce an **abstract semantics for the VQC language on intervals** of inputs (real values) and of amplitudes (complex values) in Section 5. Such a semantics provides a formal framework for reasoning on the sources of imprecision with the aim of reducing the uncertainty due to the approximation. We analyze how our semantics, built on a non-relational domain, loses precision in a relational setting like normalized quantum states, highlighting its incompleteness and **identifying sufficient conditions for completeness** (Section 6). In Section 7, we define a **VQC-based classifier** in both the concrete and abstract settings, formally characterizing the function that performs classification based on the quantum computation results. Building on this definition, we define the verification problem for VQC-based classifiers, proving its **hardness**. In Section 8 we propose some techniques to **mitigate the loss of precision** during verification. In Section 9, we **demonstrate the impact of our new verification framework** by applying it to verifying the robustness of several VQC-based classifiers on standard benchmarks also used in [41, 70], namely *Iris* [55] and *MNIST* [39] datasets. To the best of our knowledge, our verification pipeline is the first tool able to provably certify the maximum ϵ input perturbation that is tolerable by a given VQC.

2 BACKGROUND

2.1 Quantum Computation

In this section, we briefly recall the main aspects of quantum computation that make this computational model different from the classical one. For more details, we refer to [54]. In the circuit model of computation, the first difference between the two computational paradigms is that in a quantum circuit, wires represent quantum bits, or qubits, rather than bits.

According to the postulates of Quantum Mechanics, the state of one qubit is any vector of norm 1 in a two-dimensional complex Hilbert space, and the state of a register of n qubits is a vector of norm 1 in the 2^n-dimensional complex Hilbert space corresponding to the tensor product of the 2-dimensional complex Hilbert space of each qubit. The complex coefficients are called probability amplitudes for their role in the measurement process described later.

We will use the standard notation in quantum physics and indicate a vector $(\alpha_1, \ldots, \alpha_m)$ in a m-dimensional Hilbert space with respect to a basis $\mathcal{B}$ by the Dirac ket $|\psi\rangle = \sum_{k=1}^{m} \alpha_k |e_k\rangle$, where $|e_k\rangle \in \mathcal{B}$. Thus, for example, the standard[1] basis vectors of the state space of one qubit will be denoted by $|0\rangle$ and $|1\rangle$ corresponding to $[1, 0]^T$ and $[0, 1]^T$, respectively, and a generic vector, $[\alpha, \beta]^T$, in this space by $\alpha |0\rangle + \beta |1\rangle$. The postulate about the composition of quantum systems by tensor product allows us to construct the standard basis for the space of two qubits by $\{|0\rangle, |1\rangle\} \otimes \{|0\rangle, |1\rangle\} = \{|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle\}$, also denoted by $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, corresponding to (in linear algebraic notation) $\{[1, 0, 0, 0]^T, [0, 1, 0, 0]^T, [0, 0, 1, 0]^T, [0, 0, 0, 1]^T\}$. By using the same procedure, we can construct the standard basis for an m-dimensional Hilbert space.

An example of a quantum circuit is shown in Fig. 1, where the two initial gates represent two particular unitary operators on one qubit, namely H, which transforms a basis vector into a superposition, and X, which is the unitary implementation of the classical NOT gate. All operations on 1 qubit correspond to rotations of the point in the unitary three-dimensional sphere (the so-called Bloch sphere) with polar coordinates corresponding to the qubit's amplitudes. The rotations, Rx, Ry, Rz, around the three axes x, y, z are universal for all operations on one qubit, in this case H is a

---

[1]Every set of m orthonormal vectors is a basis for an m-dimensional Hilbert space. The standard basis is used as a default basis when no other definition is given.
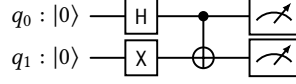
Fig. 1. A simple quantum circuit implementing the unitary operator $\mathsf{CX} \cdot (\mathsf{X} \otimes \mathsf{H})$ applied to $|00\rangle$.

rotation of a $\pi/2$ angle around the y axes while the NOT is a rotation of π around y or x. The next gate in the circuit is the controlled-NOT (CX) operation, which applies X to the second qubit only if the first qubit is $|1\rangle$. This circuit implements the unitary operator $\mathsf{CX} \cdot (\mathsf{X} \otimes \mathsf{H})$. Here, the tensor product $(\mathsf{X} \otimes \mathsf{H})$ represents the operation H applied to the less significant qubit (q_0) and X to the most significant one, q_1. If the initial state is $|00\rangle$, where bot q_0 and q_1 are $|0\rangle$, $|\psi_1\rangle = (\mathsf{X} \otimes \mathsf{H}) \cdot |00\rangle = |1\rangle \, 1/\sqrt{2}(|0\rangle + |1\rangle)$, i.e. we have negated q_1 and applied H to q_0. The final state is $\mathsf{CX} |\psi_1\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$. All quantum circuits can be constructed by using only gates in $\{\mathsf{Rx}, \mathsf{Rx}, \mathsf{Rx}, \mathsf{CX}\}$, i.e., this set is universal for quantum computation[6]. The final gate is the measurement operation, which is necessary to extract the final (classical) result from the quantum state $|\psi_2\rangle$ obtained from the quantum evolution of the input $|00\rangle$. Formally, quantum measurement corresponds to the application of projection operators[2] $\{M_e\}_{\{e \in O\}}$, where O is the set of all labels (possible outcomes) associated with the basis $B = \{|e\rangle\}_{e \in O}$ of the state space. Thus, if the system is in the quantum state $|\psi\rangle$ before the measurement, the probability of obtaining the outcome e is given by $p(e) = \|M_e |\psi\rangle\|^2$, i.e. the squared modulus of the probability amplitude (coefficient) of $|e\rangle$ in $\psi = \sum_{e \in O} \alpha_e |e\rangle$. As an example, to measure a qubit in the standard basis $\{|0\rangle, |1\rangle\}$ means to apply the projector operators M_0 on the axis $|0\rangle$ and M_1 on the axis $|1\rangle$. If the qubit state before the measurement is $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$, the probability of measuring 0 is $p(0) = \|M_0 |\psi\rangle\|^2 = |\alpha|^2$, and the probability of measuring 1 is $p(1) = \|M_1 |\psi\rangle\|^2 = |\beta|^2$.

2.2 Abstract Interpretation in a Nutshell

In the standard framework of *abstract interpretation* [19], abstract domains can be equivalently formulated either in terms of Galois connections or closure operators [20]. A pair of functions $\alpha : \mathsf{L} \to \mathsf{A}$ and $\gamma : \mathsf{A} \to \mathsf{L}$ on posets forms an *adjunction* or a *Galois insertion* (GI, denoted $\mathsf{L} \xleftrightarrow[\alpha]{\gamma} \mathsf{A}$) if for any $x \in \mathsf{L}$ and $y \in \mathsf{A}$ we have $\alpha(x) \leq_\mathsf{A} y \Leftrightarrow x \leq_\mathsf{L} \gamma(y)$. In this case, α (resp. γ) is the *left-* (resp. *right-*) *adjoint* of γ (resp. α), and it is additive (resp. co-additive). Given a co-additive[3] function γ we can always build its adjoint as $\alpha(x) = \bigwedge_\mathsf{A} \{ y \in \mathsf{A} \mid x \leq_\mathsf{L} \gamma(y) \}$, and vice-versa, given any additive function α it is always possible to build its adjoint. Abstract interpretation is generally used for abstracting *functions*. Given a function $\mathsf{f} : \mathsf{L}_i \to \mathsf{L}_o$, an input abstraction $\mathsf{L}_i \xleftrightarrow[\alpha_i]{\gamma_i} \mathsf{A}_i$ and an output abstraction $\mathsf{L}_o \xleftrightarrow[\alpha_o]{\gamma_o} \mathsf{A}_o$, we say that $\mathsf{f}^\sharp : \mathsf{A}_i \to \mathsf{A}_o$ is a sound abstraction of f if $\forall x \in \mathsf{L}_i. \, \alpha_o \circ \mathsf{f}(x) \leq_{\mathsf{L}_o} \mathsf{f}^\sharp \circ \alpha_i(x)$, meaning that we may lose information when computing abstract inputs. The abstract function is complete if $\forall x \in \mathsf{L}_i. \, \alpha_o \circ \mathsf{f}(x) = \mathsf{f}^\sharp \circ \alpha_i(x)$, i.e., when there is no loss of precision by computing with objects in the abstract domain [19, 20]. If completeness holds for a fixed sets of inputs X we say that $\mathsf{f}^\sharp$ is local complete on X [10, 46] The best of such approximation is called best correct approximation (bca for short), and it is defined as $\alpha_o \circ \mathsf{f} \circ \gamma_i$ [20]. Note that if there exists a $\mathsf{f}^\sharp$ that is complete, then also the bca is complete [19, 20, 32].

---

[2]We will consider here only a special type of quantum measurement called 'projective measurement'.
[3]A function is co-additive if it commutes with the greatest lower bound, dually an additive function commutes with the least upper bound.
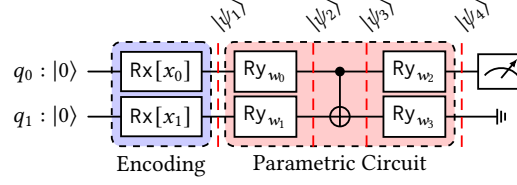
Fig. 2. A Variational Quantum Circuit. The input x_0, x_1 are encoded in a quantum state by $\mathsf{Rx}[x_0]$ and $\mathsf{Rx}[x_1]$. The weights w_i are the trainable parameters optimized during the training. The final measurement provides the values required for classical optimization.

3 VARIATIONAL QUANTUM CIRCUITS

A Variational Quantum Circuit (VQC) consists of three main components: an input encoding (feature map or quantum embedding), a parametric circuit, and a final measurement followed by a classical optimization of a cost function for the parameters' update. We describe these components via the small circuit on two qubits depicted in Fig. 2. In this example, the encoding consists of two gates $\mathsf{R_x}[x_i]$, $i \in \{0, 1\}$, which are applied to the qubits q_0 and q_1, both in the blank state $|0\rangle$. The initial state is $|00\rangle = |0\rangle_{q_1} \otimes |0\rangle_{q_0}$. These gates are rotations around the x-axis by an angle that is proportional to the input features x_i of the data and, for a generic x, are therefore defined by

$$\mathsf{Rx}[x] = \begin{pmatrix} \cos(x/2) & -\iota \sin(x/2) \\ -\iota \sin(x/2) & \cos(x/2) \end{pmatrix}, \tag{1}$$

We use the notation $\mathsf{Rx}^{q_0}[x_0]$ for the operator $\mathsf{I} \otimes \mathsf{Rx}[x_0]$, where I is the identity operator, acting as a rotation on q_0 and as the identity on q_1. Thus, $\mathsf{Rx}^{q_0}[x_0]$ is a 4×4 matrix that acts on the whole vector space of the circuit. In our example, the encoding can be obtained by computing:

$$\mathsf{Rx}^{q_1}[x_1] \cdot \mathsf{Rx}^{q_0}[x_0] \cdot |00\rangle = \cos(x_1/2)\cos(x_0/2)|00\rangle + \iota\cos(x_1/2)\sin(x_0/2)|01\rangle +$$
$$+ \iota\sin(x_1/2)\cos(x_0/2)|10\rangle - \sin(x_1/2)\sin(x_0/2)|11\rangle$$

This shows how the encoding gates map the values of x_i in the amplitude of a quantum state, on which we can now operate with the parametric quantum circuit. As a concrete instantiation, consider some classical data represented by the vector $[x_0, x_1]^T = [6.0, 2.7]^T$. The encoding phase applies $\mathsf{Rx}[6.0]$ to q_0 and $\mathsf{Rx}[2.7]$ to q_1, with $|q_0 q_1\rangle = |00\rangle$, obtaining:

$$|\psi_1\rangle = (\mathsf{Rx}^{q_1}[2.7] \cdot \mathsf{Rx}^{q_0}[6.0])\,|00\rangle = -0.22\,|00\rangle - 0.03i\,|01\rangle + 0.97\iota\,|10\rangle - 0.14\,|11\rangle \tag{2}$$

On this quantum state, we can now apply the parametric quantum circuit, which, in our example, is represented by the *ansatz* in Fig. 2 with trainable parameters $w_0, \ldots, w_3$. As with Neural Networks, during the training phase, the parameters w_i are optimized to minimize the classification error. To distinguish the rotation gates used in the input encoding phase from those in the variational part, we will denote the latter by R_w, where w is the weight optimized during training. When verifying the circuit, this is a fixed constant, as verification is performed after the training phase. In our example, the training phase produces $w_0 = 0.99$, $w_1 = -0.50$, $w_2 = 3.27$ and $w_3 = -0.69$. Thus, by first applying to $|\psi_1\rangle$ the gates $\mathsf{Ry}^{q_1}_{0.99}$, $\mathsf{Ry}^{q_0}_{-0.50}$, and then CX^{q_0, q_1} followed by $\mathsf{Ry}^{q_0}_{3.27}$ and $\mathsf{Ry}^{q_1}_{-0.69}$, we obtain the final state $|\psi_4\rangle$:

$$|\psi_4\rangle = (0.14 - 0.49\iota)\,|00\rangle - (0.11 - 0.46\iota)\,|01\rangle + (0.08 + 0.03\iota)\,|10\rangle + (0.17 + 0.70\iota)\,|11\rangle \tag{3}$$

The last step is the measurement of the qubits in the computational basis. We have four possible outcomes: 00, 01, 10, and 11. By computing the squared modulus[4] of the amplitude of each state, we

---

[4]Given a complex number $z = a + ib$, the squared modulus $|z|^2 = a^2 + b^2$.

obtain the distribution $(0.26, 0.21, 0.01, 0.52)$ over the possible outcomes $\{00, 01, 10, 11\}$. Based on this, a classification can be made by looking at the result of q_0. Since q_0 is the least significant qubit, the probability of measuring q_0 in state 1 corresponds to measuring 01 or 11 and can be computed as $0.21 + 0.52 = 0.73$. Similarly, the probability of measuring it in state 0 (i.e., measuring 00 or 10) is $0.26 + 0.01 = 0.27$. This means that we can classify the input $[x_0, x_1]^T = [6.0, 2.7]^T$ as class 1 with a probability of 0.73.

In the training of a VQC, the measurement outcome of the qubit is typically used to compute a cost function, which is then optimized by using classical optimization tools to update the parameters of the circuits. For the verification task, which is the purpose of this paper, we can concentrate on trained VQC, where the classical optimization phase has already taken place.

4 A LANGUAGE FOR VARIATIONAL QUANTUM CIRCUITS

In order to formally verify a VQC, we introduce a language, which we call $\mathcal{L}_{\text{VQC}}$, defining the structure and the behavior of VQCs. For this language, we first define a syntax (establishing the structure of a VQC), and then we give a semantics, inductively defined on the constructs of the syntax, by using a function-based representation of quantum states.

4.1 Syntax

Let q, q_i, q_j denote the VQC qubits and x a classical input variable. We define $\mathcal{L}_{\text{VQC}}$ as the language generated by the grammar:

$$
\begin{aligned}
\mathsf{E} &:= \mathsf{Rx}^q[x] \mid \mathsf{Ry}^q[x] \mid \mathsf{Rz}^q[x] && \text{(Encoding operations)} \\
\mathsf{U} &:= \mathsf{Rx}^q_w \mid \mathsf{Ry}^q_w \mid \mathsf{Rz}^q_w \mid \mathsf{CX}^{q_i, q_j} && \text{(Parametric operations)} \\
\mathsf{s} &:= \mathsf{E} \mid \mathsf{U} \mid \mathsf{s}; \mathsf{s} && \text{(Quantum statements)} \\
\mathsf{c} &:= \mathsf{s}; \mathsf{M} && \text{(VQC)}
\end{aligned}
$$

A VQC c is any element in the language $\mathcal{L}_{\text{VQC}}$ consisting of a quantum circuit s, composed by quantum statements, followed by a final measurement operation M. A quantum statement is a sequential composition of parametric and encoding operations. E represents the encoding operations, used to embed input values into quantum states. Each operation E represents a rotation around one of the three axes of the Bloch sphere, with an angle determined by the input variable x and applied to qubit q. U defines the operations used in the parametric (trainable) part of the circuit. It includes single-qubit rotations and the controlled-NOT operation CX^{q_i, q_j}, where q_i is the control and q_j the target qubit, with $q_i \neq q_j$. We syntactically distinguish between U and E to reflect their different roles within the circuit: Encoding operations depend on input variables, and are used to embed input data into quantum states; Parametric operations are parameterized by constants w, representing the trainable weights optimized during the training phase. For instance, the VQC in Fig. 2 can be written as:

$$\mathsf{Rx}^{q_0}[x_0]; \mathsf{Rx}^{q_1}[x_1]; \mathsf{Ry}^{q_0}_{w_0}; \mathsf{Ry}^{q_1}_{w_1}; \mathsf{CX}^{q_0, q_1}; \mathsf{Ry}^{q_0}_{w_2}; \mathsf{Ry}^{q_1}_{w_3}; \mathsf{M}$$

We recall that, even if we are considering only the control-not and one-qubit rotation gates, we can represent in $\mathcal{L}_{\text{VQC}}$ any possible quantum circuit [6].

4.2 Semantics

To define a semantics for $\mathcal{L}_{\text{VQC}}$, we first need to specify the *domain* of the denotations of data and then a function modeling how the computation transforms such data.

4.2.1 Semantic Data Domains. Let QV_c represent the set of qubits in a VQC $c \in \mathcal{L}_{\text{VQC}}$. Each qubit $q \in \mathsf{QV}_c$ is associated with a two-dimensional Hilbert space $\mathcal{H}_q$. Consequently, the global vector

space of the program is denoted as $\mathcal{H}_{QV_c} = \bigotimes_{q \in QV_c} \mathcal{H}_q$. The program state is then represented as a vector $|\psi\rangle \in \mathcal{H}_{QV_c}$. In this work, we use an alternative representation rather than the standard vector formalism, which is more suitable for defining the abstract domain. Since a quantum state corresponds to a superposition of basis states, each one associated with a complex amplitude, we represent quantum states ψ as mappings from the set $\mathcal{B}_c$ of standard basis vectors of $\mathcal{H}_{QV_c}$, to complex amplitudes in $\mathbb{C}$. Formally, a state $\psi \in \Psi_c \stackrel{\text{def}}{=} \mathcal{B}_c \to \mathbb{C}$ can be represented as $\psi = \lambda e \in \mathcal{B}_c. z$ ($z \in \mathbb{C}$), where the normalization condition $\sum_{e \in \mathcal{B}_c} |\psi(e)|^2 = 1$ must hold.

This functional representation is equivalent to the standard vector representation. Throughout the paper, we use ψ to denote the state as a mapping and $|\psi\rangle$ to refer to its equivalent vector representation. For instance, consider the state $|\psi_1\rangle$ defined in Equation 2. Since this is a two-qubit state, the basis set in mapping notation is $\mathcal{B}_c = \{00, 01, 10, 11\}$. Then, the corresponding mapping is: $\psi_1 = \{00 \mapsto -0.22, \ 01 \mapsto -0.03\iota, \ 10 \mapsto +0.97\iota, \ 11 \mapsto -0.14\}$.

States are not the only data on which a VQC works. Indeed, we must also consider the input for the variables x. In particular, we need to assign them real values—that is, we require an *input environment*. Given $c \in \mathcal{L}_{VQC}$, let CV_c denote the set of classical input variables. We define an input environment as a mapping $\sigma \in \Sigma_c \stackrel{\text{def}}{=} CV_c \to \mathbb{R}$, associating each variable in CV_c with a real-valued input in $\mathbb{R}$. For instance, in the example described in Section 3, the input environment is given by: $\sigma = \{x_0 \mapsto 6.0, \ x_1 \mapsto 2.7\}$.

4.2.2 Quantum Statements Semantics. We can define the semantics of a quantum statement s as a function $[\![s]\!] : \Sigma_c \to (\Psi_c \to \Psi_c)$, which, given an input environment, transforms an initial state $\psi \in \Psi_c$ into a resulting quantum state in Ψ_c.

Parametric Operations U. The semantics of every parametric operation corresponds to a matrix that describes the effect of this operation (whether a single-qubit rotation or a controlled-Not) on the full Hilbert space of the program. For example, consider the parametric operation Ry_w^q. Ry_w corresponds to a 2×2 unitary matrix acting on a single qubit. However, to represent the semantics of the rotation in the full program space, we must define a unitary matrix of size $2^{|QV_c|} \times 2^{|QV_c|}$ for Ry_w^q. This larger matrix models the action of Ry_w on qubit q while acting as the identity on all other qubits in $QV_c \setminus \{q\}$. Concretely, this is achieved by taking the tensor product of the 2×2 matrix representing Ry_w with identity matrices corresponding to the remaining qubits.

Given an operation in U, we abuse notation considering as $U \in \mathbb{C}^{2^{|QV_c|} \times 2^{|QV_c|}}$ the matrix that describes the effect of the operation in the full Hilbert space of the program. Given this matrix, each entry $U_{e_1, e_2} \in \mathbb{C}$[5] denotes the amplitude of transitioning from basis state e_2 to e_1, and applying U to a quantum state $|\psi\rangle \in \mathbb{C}^{|\mathcal{B}_c|}$ corresponds to the matrix-vector product $U \cdot |\psi\rangle$. To define the semantics of this transformation in our setting, we must express matrix multiplication using the mapping-based formalism employed to represent abstract states. Given an operation in U, its semantics is defined as a function $[\![U]\!] : \Sigma_c \to (\Psi_c \to \Psi_c)$, defined as:

$$[\![U]\!]_\sigma \stackrel{\text{def}}{=} \lambda\psi. \left(\lambda e \in \mathcal{B}_c. \sum_{e' \in \mathcal{B}_c} U_{e, e'} \cdot \psi(e') \right) \tag{4}$$

Since U is not an encoding operation, its semantics is independent of the input environment σ. For instance, consider a one qubit quantum state $|\psi\rangle_q = [a, b]^T$ (equivalently represented as $\psi_q = \{0 \mapsto a, 1 \mapsto b\}$), and the operation described by the matrix $G = \begin{bmatrix} u_{0,0} & u_{0,1} \\ u_{1,0} & u_{1,1} \end{bmatrix}$. In vector formalism the application of G to $|\psi\rangle$ is given by: $G \cdot |\psi\rangle = [u_{0,0}a + u_{0,1}b, u_{1,0}a + u_{1,1}b]^T$. that

---

[5]We recall that the bit strings e_1 and e_2 that represent standard basis elements can be used as indices to the rows and columns of the matrix U, respectively.

correspond, in the mapping notation to: $[\![\mathsf{G}]\!](\psi) = \{0 \mapsto u_{0,0}a + u_{0,1}b, 1 \mapsto u_{1,0}a + u_{1,1}b\} = \{0 \mapsto u_{0,0} \cdot \psi(0) + u_{0,1} \cdot \psi(1), 1 \mapsto u_{1,0} \cdot \psi(0) + u_{1,1} \cdot \psi(1)\}$.

Encoding Operations E. We can adopt an approach similar to the one used for parametric operations. E represents single-qubit rotations by an angle depending on the value of x. This means that every rotations are describe by a matrix over functions, $\mathsf{Fun}[x]^{2\times2}$, where $\mathsf{Fun}[x] = \{f[x] \mid f : \mathbb{R} \to \mathbb{C}, x \in \mathsf{CV_c}\}$ is a set of complex-valued functions over the variable x. The matrix $\mathsf{Rx}[x]$ in Equation 1 is an example of such a rotation in E. Abusing notation, by E we refer to the matrix modeling the encoding operation acting on the entire system: the matrix of the specific rotation on a qubit in tensor with the identity on the rest. Accordingly, $\mathsf{E} \in \mathsf{Fun}[x]^{2^{|\mathsf{QV_c}|}\times2^{|\mathsf{QV_c}|}}$.

In order to define the semantics of E, we first need to evaluate the functions $f[x] \in \mathsf{Fun}[x]$ with respect to an input environment $\sigma \in \Sigma_c$. We define the interpretation function $(\![f[x]]\!) : \Sigma_c \to \mathbb{C}$ as: $(\![f[x]]\!)_\sigma = f(\sigma(x)) \in \mathbb{C}$, that is, we evaluate the function f on the input value $\sigma(x)$. Given an encoding operation E, each matrix entry E_{e_1,e_2} is a function in $\mathsf{Fun}[x]$, mapping an input value to a complex coefficient. The semantics of E is a function $[\![\mathsf{E}]\!] : \Sigma_c \to (\Psi_c \to \Psi_c)$ defined as:

$$[\![\mathsf{E}]\!]_\sigma \overset{\text{def}}{=} \lambda\psi. (\lambda e \in \mathcal{B}_c. \sum_{e' \in \mathcal{B}_c} (\![\mathsf{E}_{e,e'}]\!)_\sigma \cdot \psi(e')) \tag{5}$$

Sequential Composition. The last operator for building a VQC is sequential composition of statements $s_1; s_2$. In this case, the semantics is trivially the functional composition of the semantics:

$$[\![s_1; s_2]\!]_\sigma \overset{\text{def}}{=} [\![s_2]\!]_\sigma \circ [\![s_1]\!]_\sigma$$

4.2.3 The Semantics of a VQC. Since a VQC $c \in \mathcal{L}_{\mathsf{VQC}}$ is built as $c := s; \mathsf{M}$, its semantics requires a denotation for the final measurement operator. This can be defined as a function that, given a quantum state, returns a probability distribution over the set of possible results. In our setting, we define $\triangle_c \overset{\text{def}}{=} \mathcal{B}_c \to \mathbb{R}$ (with metavariable ρ), the set of all possible probability distributions over the bases $\mathcal{B}_c$. The measurement semantics can be expressed (abusing notation) as a function $[\![\mathsf{M}]\!] : \Psi_c \to \triangle_c$, formally defined as:

$$[\![\mathsf{M}]\!] \overset{\text{def}}{=} \lambda\psi. (\lambda e \in \mathcal{B}_c. |\psi(e)|^2). \tag{6}$$

For instance, if we consider the state in Equation 3, that in the mapping representation is $\psi_4 = \{00 \mapsto (0.14 - 0.49\iota), 01 \mapsto -(0.11 - 0.46\iota), 10 \mapsto (0.08 + 0.03\iota), 11 \mapsto (0.17 + 0.70\iota)\}$. The result of the application measurement operators is:

$$[\![\mathsf{M}]\!](\psi_4) = \{00 \mapsto 0.26, 01 \mapsto 0.21, 10 \mapsto 0.01, 11 \mapsto 0.52\} \in \triangle_{q_0,q_1}$$

Finally, the concrete semantics of the entire $c = s; \mathsf{M}$, is the composition of the semantics of the measurement applied to the result of operator s semantics, obtaining (again abusing notation) a function $[\![c]\!]: \Sigma_c \to (\Psi_c \to \triangle_c)$ defined as:

$$[\![c]\!]_\sigma \overset{\text{def}}{=} \lambda\psi. [\![\mathsf{M}]\!] \circ [\![s]\!]_\sigma(\psi).$$

When verifying a VQC, we execute the circuit on a set of input values, which corresponds to a specific input and all possible values obtained by perturbing it. In the concrete domain, this would involve executing the circuit for every possible input. To formalize this, we define the concrete semantics of the circuit in a collecting style, i.e., as a function $[\![c]\!] : \wp(\Sigma_c) \to (\wp(\Psi_c) \to \wp(\triangle_c))$. We abuse notation by using the same semantic notation for the additive lift of $[\![\cdot]\!]$ to sets of environments and states, i.e., $[\![c]\!]_\Sigma(\Psi) \overset{\text{def}}{=} \{ [\![c]\!]_\sigma(\psi) \mid \sigma \in \Sigma, \psi \in \Psi \}$, where $\Sigma \in \wp(\Sigma_c)$ and $\Psi \in \wp(\Psi_c)$ represent a set of initial states of the VQC.

5 ABSTRACTING $\mathcal{L}_{\text{VQC}}$ SEMANTICS

In many fields of computation, the concrete semantics is usually an intractable model of computation due to computability or computational complexity reasons. In this section, we follow the idea widely exploited both in programming languages and in NN [20, 30], consisting of executing the computational system on *approximated data*, e.g., intervals of numerical values. In order to approximate the computation, we first need to abstract the data. We observe that inputs (environments) and outputs (distributions) are mappings to real values, while the VQC works on states, which are mappings to complex values. This means that we have to split the abstraction process, namely, we have to introduce the real intervals abstraction for environments and distributions, and the complex interval abstraction for the states.

5.1 Abstracting Environments (inputs) and Distributions (outputs)

Let us first extend the standard abstract domain of intervals to real values, with a slight change consisting of considering only bounded intervals.

5.1.1 Reals Interval Domain. Let us consider the domain of closed [18] and bounded [57] intervals on $\mathbb{R}$, i.e., $\mathbb{RI} \stackrel{\text{def}}{=} \{ [r_l, r_u] \mid r_l, r_u \in \mathbb{R} \} \cup \{ \varnothing, \mathbb{R} \}$. As it happens for integer intervals, also this domain can be characterized as an abstraction of the powerset of reals $\langle \wp(\mathbb{R}), \subseteq \rangle$ [18]. We define two functions $\alpha_{\text{RI}} : \wp(\mathbb{R}) \to \mathbb{RI}$ and $\gamma_{\text{RI}} : \mathbb{RI} \to \wp(\mathbb{R})$ as: Let $I \in \mathbb{RI}$

$$\gamma_{\text{RI}}(I) \stackrel{\text{def}}{=} \begin{cases} \{ r \mid r_l \leq r \leq r_u \} & \text{if } I = [r_l, r_u] \\ I & \text{if } I = \varnothing \text{ or } I = \mathbb{R} \end{cases} \qquad \alpha_{\text{RI}}(X) \stackrel{\text{def}}{=} \bigcap \{ I \in \mathbb{RI} \mid X \subseteq \gamma_{\text{RI}}(I) \}$$

Note that the definition of α_{RI} is a good definition since closed intervals on $\mathbb{R}$ are closed sets in the corresponding topological space, and the intersection of any family of closed sets is a closed set, with $\varnothing$ and $\mathbb{R}$ closed sets. In general, let $X \in \wp(\mathbb{R})$, $\alpha_{\text{RI}}(X) = [\min(X), \max(X)]$ if both $\min(X)$ and $\max(X)$ exist in X; If at least one of them does not exist since X has an *open* bound, e.g., $X = \{ r \mid 2 < r \leq 10 \}$, then the abstraction closes it, namely $\alpha_{\text{RI}}(X) = [2, 10]$; If, at least one bound not exist since the values in X grow to infinite (or decrease to negative infinite), e.g., $X = \{ r \in \mathbb{R} \mid r \geq 10 \}$, then $\alpha_{\text{RI}}(X) = \mathbb{R}$; If $X = \varnothing$ then the abstraction is $\varnothing$. When $\alpha_{\text{RI}}(X) = [l, u]$ we abuse notation writing $l = \min(X)$ and $u = \max(X)$ [57].

The following results come from the fact that γ_{RI} is a co-additive function over $\mathbb{RI}$, and therefore, by construction, α_{RI} is its adjoint function forming a Galois Insertion.

PROPOSITION 5.1. $\langle \wp(\mathbb{R}), \subseteq \rangle \xleftarrow[\alpha_{\text{RI}}]{\gamma_{\text{RI}}} \langle \mathbb{RI}, \leq_{\mathbb{RI}} \rangle$, *where $\leq_{\mathbb{RI}}$ is standard inclusion between intervals.*

In [18, 51] we can find the whole arithmetic on real intervals. Here we will use only multiplication (extended to matrices) and sum, that we denote, abusing notation, as $+$ and $\cdot$.

5.1.2 Abstract Environments. Environments map input variables to real values. In order to abstract the VQC computation to intervals, we have to define *abstract environments* $\overline{\mathsf{S}}$ mapping the input variables of $\mathsf{c} \in \mathcal{L}_{\text{VQC}}$ (i.e., those in CV_{c}) to the *intervals* of values in $\mathbb{RI}$ representing the range of possible values for that variable. This abstraction is performed on the collection of reals associated with each variable (due to a set of possible input environments). For example, if a variable $x \in \mathsf{CV}_{\mathsf{c}}$ can take values between 0 and $r \in \mathbb{R}$, the abstract environment would map x to the interval $[0, r]$. Formally, $\overline{\mathsf{S}} \in \overline{\mathsf{S}}_{\mathsf{c}} \stackrel{\text{def}}{=} \mathsf{CV}_{\mathsf{c}} \to \mathbb{RI}$ can be defined by abstract interpretation on the power domain of environments. Let $\alpha_{\text{env}} : \wp(\overline{\Sigma}_{\mathsf{c}}) \to \overline{\mathsf{S}}_{\mathsf{c}}$ defined as follows: Let $\Sigma \in \wp(\overline{\Sigma}_{\mathsf{c}})$

$$\alpha_{\text{env}}(\Sigma) \stackrel{\text{def}}{=} \lambda x \in \mathsf{CV}_{\mathsf{c}}. \, \alpha_{\text{RI}}(\{ \sigma(x) \mid \sigma \in \Sigma \}),$$

This function abstracts a set of inputs (i.e., a set of static environments) into a single abstract environment $\overline{\mathsf{S}}$. The concretization is the standard adjoint $\gamma_{\mathsf{env}} : \overline{\mathbb{S}}_{\mathsf{c}} \rightarrow \wp(\mathbb{\Sigma}_{\mathsf{c}})$, namely $\forall \overline{\mathsf{S}} \in \overline{\mathbb{S}}_{\mathsf{c}}$ we have $\gamma_{\mathsf{env}}(\overline{\mathsf{S}}) \stackrel{\mathrm{def}}{=} \left\{ \sigma \in \mathbb{\Sigma}_{\mathsf{c}} \, \middle| \, \forall x \in \mathsf{CV}_{\mathsf{c}}. \ \sigma(x) \in \gamma_{\mathsf{RI}}(\overline{\mathsf{S}}(x)) \right\}$.

PROPOSITION 5.2. $\langle \wp(\mathbb{\Sigma}_c), \subseteq \rangle \xleftrightarrow[\alpha_{\mathsf{env}}]{\gamma_{\mathsf{env}}} \langle \overline{\mathbb{S}}_c, \dot{\leq}_{\mathbb{RI}} \rangle$[6].

5.1.3 Abstract Distributions. The output of the VQC semantics is a probability distribution, which must also be abstracted in order to return intervals of probabilities when measuring an abstract state. Formally, let us define $\overline{\mathbb{D}}_{\mathsf{c}} \stackrel{\mathrm{def}}{=} \mathcal{B}_{\mathsf{c}} \rightarrow \mathbb{RI}$, the abstract domain of probability distributions on intervals, point-wise ordered by $\dot{\leq}_{\mathbb{RI}}$. We can define the abstraction function $\alpha_{\mathsf{dist}} : \wp(\mathbb{\Delta}_{\mathsf{c}}) \rightarrow \overline{\mathbb{D}}_{\mathsf{c}}$, between sets of probability distributions $\langle \wp(\mathbb{\Delta}_{\mathsf{c}}), \subseteq \rangle$ and abstract interval distributions $\langle \overline{\mathbb{D}}_{\mathsf{c}}, \dot{\leq}_{\mathbb{RI}} \rangle$. Let $\Delta \in \wp(\mathbb{\Delta}_{\mathsf{c}})$,

$$\alpha_{\mathsf{dist}}(\Delta) \stackrel{\mathrm{def}}{=} \lambda e \in \mathcal{B}_{\mathsf{c}}. \ \alpha_{\mathsf{RI}}\left(\left\{ \rho(e) \, \middle| \, \rho \in \Delta \right\}\right)$$

The concretisation function $\gamma_{\mathsf{dist}} : \overline{\mathbb{D}}_{\mathsf{c}} \rightarrow \wp(\mathbb{\Delta}_{\mathsf{c}})$ is the standard adjoint, i.e., $\forall \varrho^{\sharp} \in \overline{\mathbb{D}}_{\mathsf{c}}. \ \gamma_{\mathsf{dist}}(\varrho^{\sharp}) \stackrel{\mathrm{def}}{=} \left\{ \rho \in \mathbb{\Delta}_{\mathsf{c}} \, \middle| \, \forall e \in \mathcal{B}_{\mathsf{c}}. \ \rho(e) \in \gamma_{\mathsf{RI}}(\varrho^{\sharp}(e)) \right\}$.

For instance, the mapping $\varrho^{\sharp} = \{00 \mapsto [0.126, 0.50], 01 \mapsto [0.116, 0.460], 10 \mapsto [0, 0.140], 11 \mapsto [0.291, 0.761]\}$ in Equation 8, express the abstract state where we can obtain 00 with probability (w.p.) $[0.126, 0.50]$, 01 w.p. $[0.116, 0.460]$, 10 w.p. $[0, 0.140]$, and 11 w.p. $[0.291, 0.761]$.

5.2 Abstracting Quantum States

In order to abstract states, mapping quantum variables to complex values, into the interval domain, we need to extend the abstract domain of intervals to complex values, i.e., to pairs of real intervals.

5.2.1 Complex Interval Domain. Let us consider the set of complex numbers $\mathbb{C}$. Given a generic complex number $z = r_1 + \iota r_2 \in \mathbb{C}$, we denote by $\mathrm{Re}(z) = r_1$ its real part, and by $\mathrm{Im}(z) = r_2$ its imaginary part. Analogously, given a $Z \in \wp(\mathbb{C})$, $\mathrm{Re}(Z) \stackrel{\mathrm{def}}{=} \left\{ \mathrm{Re}(z) \, \middle| \, z \in Z \right\}$ and $\mathrm{Im}(Z) \stackrel{\mathrm{def}}{=} \left\{ \mathrm{Im}(z) \, \middle| \, z \in Z \right\}$. Now, we can abstract complex numbers to intervals simply by keeping the information that z is determined by the pairs of real values r_1 and r_2. Following this idea, we define the abstract domain of complex intervals as $\mathbb{CI} \stackrel{\mathrm{def}}{=} \mathbb{RI} \times \mathbb{RI}$. Hence, we can trivially extend the interval abstraction and concretization functions to complex numbers as: Let $Z \in \wp(\mathbb{C})$ and $R, I \in \mathbb{RI}$

$$\alpha_{\mathsf{CI}} : \wp(\mathbb{C}) \rightarrow \mathbb{CI}, \ \alpha_{\mathsf{CI}}(Z) \stackrel{\mathrm{def}}{=} \langle \alpha_{\mathsf{RI}}(\mathrm{Re}(Z)), \alpha_{\mathsf{RI}}(\mathrm{Im}(Z)) \rangle$$

$$\gamma_{\mathsf{CI}} : \mathbb{CI} \rightarrow \wp(\mathbb{C}), \ \gamma_{\mathsf{CI}}(R, I) \stackrel{\mathrm{def}}{=} \left\{ z \in \mathbb{C} \, \middle| \, \mathrm{Re}(z) \in \gamma_{\mathsf{RI}}(R), \ \mathrm{Im}(z) \in \gamma_{\mathsf{RI}}(I) \right\}$$

PROPOSITION 5.3. $\langle \wp(\mathbb{C}), \subseteq \rangle \xleftrightarrow[\alpha_{\mathsf{CI}}]{\gamma_{\mathsf{CI}}} \langle \mathbb{CI}, \leq_{\mathbb{CI}} \rangle$, where $\forall \langle R_1, I_1 \rangle, \langle R_2, I_2 \rangle \in \mathbb{CI}$ we have $\langle R_1, I_1 \rangle \leq_{\mathbb{CI}} \langle R_2, I_2 \rangle$ iff $R_1 \leq_{\mathbb{RI}} R_2$ and $I_1 \leq_{\mathbb{RI}} I_2$.

We abuse notation by using the same operator introduced on real intervals also on complex intervals, with the only difference of being point-wise applied to the elements of the pair, i.e., $\langle [a_1, b_1], [c_1, d_1] \rangle + \langle [a_2, b_2], [c_2, d_2] \rangle = \langle [a_1, b_1] + [a_2, b_2], [c_1, d_1] + [c_2, d_2] \rangle$, analogous for $\cdot$ [29, 51].

5.2.2 Abstract States. Similarly to what we did for the abstract environments, we now need to map each standard basis vector to an interval of complex values, in order to abstract states. We define the set of abstract states $\overline{\mathbb{V}}_{\mathsf{c}} = \mathcal{B}_{\mathsf{c}} \rightarrow \mathbb{CI}$ mapping each standard basis $e \in \mathcal{B}_{\mathsf{c}}$ to a complex abstract interval $\langle [a, a'], [b, b'] \rangle \in \mathbb{CI}$. This domain $\overline{\mathbb{V}}_{\mathsf{c}}$ can be obtained as the abstraction function $\alpha_{\mathsf{st}} : \wp(\mathbb{\Psi}_{\mathsf{c}}) \rightarrow \overline{\mathbb{V}}_{\mathsf{c}}$: Let $\Psi \in \wp(\mathbb{\Psi}_{\mathsf{c}})$

$$\alpha_{\mathsf{st}}(\Psi) \stackrel{\mathrm{def}}{=} \lambda e \in \mathcal{B}_{\mathsf{c}}. \ \alpha_{\mathsf{CI}}\left(\left\{ \psi(e) \, \middle| \, \psi \in \Psi \right\}\right)$$

---

[6]$\dot{\leq}_{\mathbb{RI}}$ is the point-wise ordering induced by $\leq_{\mathbb{RI}}$ on $\overline{\mathbb{S}}_{\mathsf{c}}$, i.e., $\forall \overline{\mathsf{S}}_1, \overline{\mathsf{S}}_2 \in \overline{\mathbb{S}}_{\mathsf{c}}$ we have $\overline{\mathsf{S}}_1 \dot{\leq}_{\mathbb{RI}} \overline{\mathsf{S}}_2$ iff $\forall x \in \mathsf{CV}_{\mathsf{c}}. \ \overline{\mathsf{S}}_1(x) \leq_{\mathbb{RI}} \overline{\mathsf{S}}_2(x)$.

The concretisation function $\gamma_{\mathsf{st}} : \overline{\mathbb{V}}_c \to \wp(\Psi_c)$ is the standard adjoint, i.e., is defined as $\forall \overline{V} \in \overline{\mathbb{V}}_c. \gamma_{\mathsf{st}}(\overline{V}) \stackrel{\text{def}}{=} \{ \psi \in \Psi_c \mid \forall e \in \mathcal{B}_c . \psi(e) \in \gamma_{\mathbb{CI}}(\overline{V}(e)) \}$.

PROPOSITION 5.4. $\langle \wp(\Psi_c), \subseteq \rangle \xleftarrow[\alpha_{\mathsf{st}}]{\gamma_{\mathsf{st}}} \langle \overline{\mathbb{V}}_c, \dot{\leq}_{\mathbb{CI}} \rangle$ ($\dot{\leq}_{\mathbb{CI}}$ is the point-wise ordering induced by $\leq_{\mathbb{CI}}$).

An example of an abstract state is in Equation 7. Here, $\overline{V}_4(00) = \langle [-0.159, 0.433], [-0.559, -0.355] \rangle$, meaning that the real part of the amplitude associated with 00 ranges between -0.159 and 0.433 while the imaginary part ranges between -0.559 and -0.355.

5.3 Abstract Semantics of $\mathcal{L}_{\mathsf{VQC}}$

Starting from an abstract environment $\overline{S}$ and an abstract state $\overline{V}$, both working on intervals, we can define the abstract semantics of a VQC $c \in \mathcal{L}_{\mathsf{VQC}}$ executing c on intervals.

We first have to define an abstract evaluation of functions $f[x] \in \mathsf{Fun}[x]$ w.r.t. the abstract environment, and therefore we have to compute f on intervals. Let $\overline{S} \in \overline{\mathbb{S}}_c$, we define $(\!|f[x]|\!)^{\sharp} : \overline{\mathbb{S}}_c \to \mathbb{CI}$ as an approximation of f on intervals: $(\!|f[x]|\!)^{\sharp}_{\overline{S}} \stackrel{\text{def}}{=} \alpha_{\mathbb{CI}} \circ f \circ \gamma_{\mathbb{CI}}(\overline{S}(x)) \in \mathbb{CI}$. This is the best correct approximation of (the additive lift to sets of) f on $\mathbb{CI}$.

Given $c = s ; M \in \mathcal{L}_{\mathsf{VQC}}$, in order to define its abstract semantics, we need to define separately the abstract semantics of quantum statements s and of the measurement operator M.

Definition 5.5 (Abstract Quantum Statements Semantics). For each statement s, the abstract semantics is the function $[\![s]\!]^{\sharp} : \overline{\mathbb{S}}_c \to (\overline{\mathbb{V}}_c \to \overline{\mathbb{V}}_c)$ is defined as follows: Let $\overline{V} \in \overline{\mathbb{V}}_c$

(1) $[\![U]\!]^{\sharp}_{\overline{S}} \stackrel{\text{def}}{=} \lambda \overline{V}. \left(\lambda e \in \mathcal{B}_c. \sum_{e' \in \mathcal{B}_c} (U_{e,e'} \cdot \overline{V}(e')) \right)$

(2) $[\![E]\!]^{\sharp}_{\overline{S}} \stackrel{\text{def}}{=} \lambda \overline{V}. (\lambda e \in \mathcal{B}_c. \sum_{e' \in \mathcal{B}_c} (\!|E_{e,e'}|\!)^{\sharp}_{\overline{S}} \cdot \overline{V}(e'))$;

(3) $[\![s_1 ; s_2]\!]^{\sharp}_{\overline{S}} \stackrel{\text{def}}{=} [\![s_2]\!]^{\sharp}_{\overline{S}} \circ [\![s_1]\!]^{\sharp}_{\overline{S}}$,

Since $E_{e,e'}$ is always a sum of sine and cosine, $(\!|E_{e,e'}|\!)^{\sharp}_{\overline{S}}$ can be defined using the standard interval arithmetic [51].

Then, let M be a measurement operator, the abstract semantics of M is a function $[\![M]\!]^{\sharp} : \overline{\mathbb{V}}_c \to \overline{\mathbb{D}}_c$ defined as:

$$[\![M]\!]^{\sharp} \stackrel{\text{def}}{=} \lambda \overline{V}. (\lambda e \in \mathcal{B}_c. |\overline{V}(e)|^2)$$

where $| \cdot |^2$ is computed according to the interval arithmetic [51].

Finally, the abstract semantics of $c := s ; M$ is a function $[\![c]\!]^{\sharp} : \overline{\mathbb{S}}_c \to (\overline{\mathbb{V}}_c \to \overline{\mathbb{D}}_c)$ defined as:

$$[\![c]\!]^{\sharp}_{\overline{S}} \stackrel{\text{def}}{=} \lambda \overline{V}. [\![M]\!]^{\sharp} \circ [\![s]\!]^{\sharp}_{\overline{S}}(\overline{V}).$$

The following result tells us that computing a VQC on the abstract domain of intervals by using the so far defined abstract semantics $[\![\cdot]\!]^{\sharp}$ preserves all the concrete computations.

THEOREM 5.6 (SOUNDNESS). *The abstract VQC semantics $[\![\cdot]\!]^{\sharp}$ is sound. Formally, $\forall \Sigma \in \wp(\Sigma_c)$ and $\forall \Psi \in \wp(\Psi_c)$, then $\alpha_{\mathsf{dist}}([\![c]\!]_{\Sigma}(\Psi)) \dot{\leq}_{\mathbb{RI}} [\![c]\!]^{\sharp}_{\alpha_{\mathsf{env}(\Sigma)}}(\alpha_{\mathsf{st}}(\Psi))$*

PROOF. The soundness comes directly from the fact that the abstract semantics corresponds to a composition of operations in the interval arithmetic, and these operations are all sound. □

5.4 An Example

We consider the same VQC in Section 3. Now, suppose we want to execute the circuit for all inputs generated by perturbing the original input $[x_0, x_1]^T = [6.0, 2.7]^T$ by an $\epsilon = 0.5$. This perturbation will create a the set of environments $\Sigma = \{ \sigma \mid 5.5 < \sigma(x_0) < 6.5 \wedge 2.2 < \sigma(x_1) < 3.2 \}$. To do

this, we first define the abstract environment as $\alpha_{\mathrm{env}}(\Sigma) = \overline{S} : \{x_0 \to [5.5, 6.5], x_1 \to [2.2, 3.2]\}$. Now we can execute our circuit abstractly. We start from the initial abstract state $\overline{V}_0 = \{00 \to \langle [1, 1], [0, 0]\rangle, (01, 10, 11) \to \langle [0, 0], [0, 0]\rangle\}$, which represents the abstraction of the concrete initial state ψ_0 in Section 3. The first step is to execute the abstract semantics of the encoding. We need to evaluate the abstract semantics of $\mathrm{Rx}^{q_0}[x_0]$ and $\mathrm{Rx}^{q_1}[x_1]$ whit $\overline{S}(x_0) = [5.5, 6.5]$ and $\overline{S}(x_1) = [2.2, 3.2]$, that corresponds to:

$$\overline{V}_1 = [\![\mathrm{Rx}^{q_1}[x_1]]\!]_{\overline{S}}^{\sharp} \circ [\![\mathrm{Rx}^{q_0}[x_0]]\!]_{\overline{S}}^{\sharp}(\overline{V}_0) = \begin{cases} 00 \mapsto \langle [-0.454, 0.029], [0, 0]\rangle & 01 \mapsto \langle [0, 0], [-0.173, 0.049]\rangle \\ 10 \mapsto \langle [0, 0], [0.824, 1.0]\rangle & 11 \mapsto \langle [-0.382, 0.108], [0, 0]\rangle \end{cases}.$$

At this point, we proceed by applying the variational part of the circuit, which computes the classification. As in the concrete case, this is obtained applying, in sequence, the following operations $\mathrm{Ry}_{-0.50}^{q_0}, \mathrm{Ry}_{0.99}^{q_1}, \mathrm{CX}^{q_0, q_1}, \mathrm{Ry}_{3.27}^{q_0}$ and $\mathrm{Ry}_{-0.69}^{q_1}$. The final state is thus obtained by computing:

$$\overline{V}_4 = [\![\mathrm{Ry}_{-0.69}^{q_1}]\!]_{\overline{S}}^{\sharp} \circ [\![\mathrm{Ry}_{3.27}^{q_0}]\!]_{\overline{S}}^{\sharp} \circ [\![\mathrm{CX}^{q_0, q_1}]\!]_{\overline{S}}^{\sharp} \circ [\![\mathrm{Ry}_{0.99}^{q_1}]\!]_{\overline{S}}^{\sharp} \circ [\![\mathrm{Ry}_{-0.50}^{q_0}]\!]_{\overline{S}}^{\sharp}(\overline{V}_1) =$$

$$= \begin{cases} 00 \mapsto \langle [-0.159, 0.433], [-0.559, -0.355]\rangle & 01 \mapsto \langle [-0.404, 0.184], [0.341, 0.544]\rangle \\ 10 \mapsto \langle [-0.171, 0.328], [-0.074, 0.181]\rangle & 11 \mapsto \langle [-0.086, 0.413], [0.54, 0.768]\rangle \end{cases}. \qquad (7)$$

The last step is the measurement. We compute $[\![\mathrm{M}]\!]^{\sharp}(\overline{V}_4)$ obtaining the following abstract distribution:

$$\varrho^{\sharp} = \begin{cases} 00 \mapsto [0.126, 0.50] & 01 \mapsto [0.116, 0.460] \\ 10 \mapsto [0, 0.140] & 11 \mapsto [0.291, 0.761] \end{cases} \qquad (8)$$

As we see in Fig. 2, we consider only the measurement of q_0. Thus, as in the concrete state we sum the probability intervals of 00 and 10, obtaining that the probability of measuring 0 is $[0.126, 0.640]$ and we sum the probability of 01 and 11 obtaining that the probability of measuring 1 ranges between $[0.407, 1, 221]$. The fact that the probability of measuring 1 exceeds 1 is due to the approximations of our abstraction.

6 ON THE PRECISION OF $\mathcal{L}_{\mathrm{VQC}}$ ABSTRACT SEMANTICS

In the previous section, we have introduced a *sound* abstract semantics. Soundness means that the abstract computation may add imprecision, potentially over-approximating the concrete results. In our case, this manifests as computing intervals that are larger than those we would obtain by abstracting the result of a concrete execution.

6.1 Sources of incompleteness

In this section, we discuss where and how the abstract computation of a VQC loses precision with respect to the concrete one. To better understand the source of imprecision, we appeal to the concept of *completeness* [19, 20] of the abstraction. Analyzing completeness allows us to identify whether the loss of precision stems from the abstraction's loss of information or from the computations over the abstract domain.

LEMMA 6.1. *Let* $\mathrm{f} : \mathbb{R}^n \to \mathbb{R}$ *be defined as* $\mathrm{f}(\langle r_i\rangle_{i=1}^n) = \sum_{i=1}^n r_i$, *where* $\langle r_i\rangle_{i=1}^n$ *denotes a tuples of n reals. The abstraction of* f *over the interval domain is not (globally) complete.*

PROOF. We proceeded to find counterexamples for completeness. Let us recall that $\mathrm{f}^{\sharp} : \mathbb{RI}^n \to \mathbb{RI}$ and it is defined using the interval sum. Let $X \in \wp(\mathbb{R}^n)$ be a set of tuples, we write X_j as the set $\{r_j \mid \forall \langle r_i\rangle_{i=1}^n \in X. r_j \in \langle r_i\rangle_{i=1}^n\}$, i.e., the all set elements in position j in the tuples in R. We call $\alpha_{\mathrm{RI}^n} : \wp(\mathbb{R}^n) \to \mathbb{RI}^n$ the abstraction from a set of tuples to an interval tuple defined as: $\alpha_{\mathrm{RI}^n}(X) = \langle \alpha_{\mathrm{RI}}(X_j)\rangle_{j=1}^n$. Let $X = \{\langle r_i\rangle_{i=1}^n \mid \sum_{i=1}^n r_i^2 = 1\}$ a set of normalized reals tuples such that $\overline{X} = \alpha_{\mathrm{RI}^n}(X) = \langle [-1, 1]\rangle_{j=1}^n$. Thus $\mathrm{f}^{\sharp}(\overline{X}) = \sum_{i=1}^n [-1, 1] = [-n, n]$. On the other hand, the
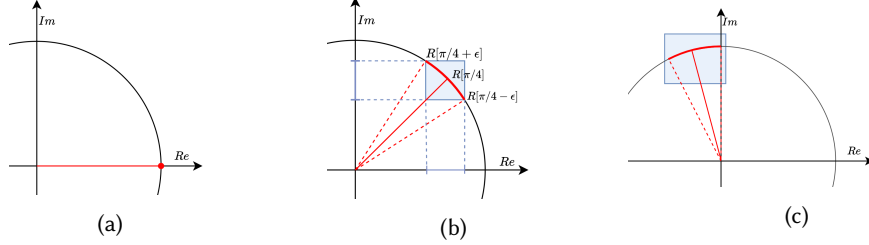
Fig. 3. We represent a complex number as a point with real and imaginary parts on the x and y axes. In (a) we represent $z = \langle 1, 0 \rangle$; in (b) the results of computing the rotation $R[\theta](z)$ with $\theta \in [\frac{\pi}{4} - \epsilon, \frac{\pi}{4} + \epsilon]$ abstractly (blue box) and point wise (red arc); in (c) the results of the abstract and point wise application of $R[\pi/3]$ to the abstract and concrete state in (b) respectively. The results in the figures use $\epsilon \approx 0.2$.

concrete image of X through f is: $f(X) = \left\{ \sum_{i=1}^n r_i \mid \langle r_i \rangle_{i=1}^n \in X \right\}$, and by the Cauchy-Schwarz inequality and the normalization constraint: $\left| \sum_{i=1}^n r_i \right| \leq \sqrt{n} \cdot \left(\sum_{i=1}^n r_i^2 \right)^{1/2} = \sqrt{n}$. Hence: $\alpha_{\mathrm{RI}}(f(X)) \leq_{\mathbb{RI}} [-\sqrt{n}, \sqrt{n}] <_{\mathbb{RI}} [-n, n] = f^\sharp(\alpha_{\mathrm{RI}^n}(X))$ i.e., the abstraction is not complete. □

From the previous lemma, we derive a more general result. The counterexample was based on tuples of normalized real numbers, i.e., $\sum r_i^2 = 1$. As shown in Equation 4 and Equation 5, the semantics of our quantum operators rely on sums of products of complex numbers. Since addition and multiplication over $\mathbb{C}$ decompose into sum of real numbers, the semantics reduces to real-valued functions over tuples of normalized values,[7] and by the previous lemma, incompleteness arises in such settings.

THEOREM 6.2 (INCOMPLETENESS). *The abstract semantics $[\![\cdot]\!]^\sharp$ of $\mathcal{L}_{\mathrm{VQC}}$ is not complete.*

We show incompleteness by means of the following example.

Example 6.3. Let us consider the following circuit on one qubit q: $\mathsf{s} := \mathsf{Rx}^q[x]; \mathsf{Rx}^q_{\pi/2}$, on the set of inputs $\Sigma : \left\{ \sigma \mid \sigma(x) \in [\pi/2 - 0.2, \pi/2 + 0.2] \right\}$. We start from initial state $\psi_0 = \{0 \mapsto 1, 1 \mapsto 0\}$. First, we compute the abstract semantics of the first rotation gate:

$$\overline{V}_1 = [\![\mathsf{Rx}^q[x]]\!]^\sharp_{\overline{S}}(\alpha_{\mathrm{st}}(\{\psi_0\})) = \{0 \mapsto ([0.51, 0.86], [0, 0]), 1 \mapsto ([0, 0], [-0.86, -0.51])\}.$$

Then, applying the $\mathsf{Rx}^q_{\pi/2}$ rotation, we obtain:

$$\overline{V}_f = [\![\mathsf{Rx}^q_{\pi/2}]\!]^\sharp_{\overline{S}}(\overline{V}_1) = \{0 \mapsto ([-0.247, 0.247], [0, 0]), 1 \mapsto ([0, 0], [-1.216, -0.722])\}.$$

We observe that abstraction introduces components with norms greater than one, since the imaginary component of the amplitude associated with basis state 1 exceeds 1, clearly values that result from over-approximation noise. Thus this examples shows that, given $\psi_0 = \{0 \rightarrow 1, 1 \rightarrow 0\}$, $\alpha_{\mathrm{CI}} \circ [\![\mathsf{s}]\!]_\Sigma(\{\psi_0\}) \dot{<}_{\mathrm{CI}} [\![\mathsf{s}]\!]^\sharp_{\alpha_{\mathrm{env}}(\Sigma)}(\alpha_{\mathrm{st}}(\{\psi_0\}))$, i.e. the abstract semantics is incomplete.

We conclude that the non-relational nature of the interval domain introduces significant over-approximation, especially in the quantum setting, where amplitudes and matrix coefficients are inherently dependent due to normalization and unitary constraints.

A visual example of how overapproximation arises in the 2D complex plane is shown in Fig. 3. We observe that applying a rotation to a point generates an arc on the unit circle. The interval abstraction, which treats each component independently, encloses this arc within a bounding box

---

[7]Both state vectors ψ and matrix coefficients are normalized, due to unitarity and state normalization constraints.

that may include points not present in the actual set. While the first abstraction in Fig. 3b, is complete (the box exactly contains the arc without including any additional points on the circumference), the situation changes after a second rotation (Fig. 3c): due to the extra points already introduced by the first abstraction, the new bounding box now includes not only the concrete points in the rotated arc but also additional, spurious points along the unit circle causing incompleteness.

6.2 On the Completeness of the Abstract Semantics

We have seen that completeness cannot hold for all possible inputs, i.e., it fails *globally* [20, 32]. However, we can still investigate whether completeness holds under certain conditions on the operators or by restricting the set of possible inputs.[8] This perspective allows us to better understand when and where precision is lost, namely, which inputs and which computations are responsible.

As shown in Equation 4 and Equation 5, the semantics of the statements involve sums of complex numbers that are linked by normalization constraints, which ultimately cause incompleteness. However, as we are about to show, by restricting either the type of operations or the structure of the inputs, these sums over complex numbers may simplify to single terms. This would eliminate the dependency issues discussed in the previous section and ensure completeness.

THEOREM 6.4. *If* U *is a* generalized permutation[9], $[\![\mathsf{U}]\!]^{\sharp}$ *is complete.*

PROOF. Given a generalized permutation U, for each row of the matrix, there is only one entry different from zero. We recall that given a basis $e \in \mathcal{B}_c$, the entries $\{\mathsf{U}_{e,e'}\}_{\forall e' \in \mathcal{B}_c}$ corresponds to one row of the matrix U. Thus for each e there is one base u_e that corresponding to only non-zero entry in the row e, in other words, $\forall e \in \mathcal{B}_c$. $\mathsf{U}_{e,e'} \neq 0$ iff $e' = u_e$. Thus, considering the equations in Equation 4, if U is a generalized permutation its semantics is $[\![\mathsf{U}]\!]_{\Sigma} = \lambda\psi. \left(\lambda e \in \mathcal{B}_c.(\mathsf{U}_{e,u_e} \cdot \psi(u_e))\right)$, where u_e is the index of the only non-zero entry for each row e. Similarly, considering the Definition 5.5, we have $[\![\mathsf{U}]\!]_{\overline{\mathsf{S}}}^{\sharp} = \lambda\overline{\mathsf{V}}. \left(\lambda e \in \mathcal{B}_c.(\mathsf{U}_{e,u_e} \cdot \overline{\mathsf{V}}(u_e)\right)$. According to these simplifications, $[\![\mathsf{U}]\!]^{\sharp}$ is complete, i.e., $\forall \Psi \in \wp(\Psi_c), \Sigma \in \wp(\Sigma_c)$ $\alpha_{\mathrm{st}}([\![\mathsf{U}]\!]_{\Sigma}(\Psi)) = [\![\mathsf{S}]\!]_{\alpha_{\mathrm{env}}(\Sigma)}^{\sharp}(\alpha_{\mathrm{st}}(\Psi))$, if and only if $\alpha_{\mathrm{st}}(\{ \lambda e \in \mathcal{B}_c.(\mathsf{U}_{e,u_e}) \cdot \psi(u_e) \mid \psi \in \Psi \}) = \lambda e \in \mathcal{B}_c.(\mathsf{U}_{e,u_e} \cdot (\alpha_{\mathrm{st}}(\Psi))(u_e)$ that holds iff $\forall e \in \mathcal{B}_c$. $\alpha_{\mathrm{CI}} \left(\{ (\mathsf{U}_{e,u_e} \cdot \psi(u_e)) \mid \psi \in \Psi \}\right) = \left(\mathsf{U}_{e,u_e} \cdot \alpha_{\mathrm{CI}} \left(\{ \psi(u_e) \mid \psi \in \Psi \}\right)\right)$. Since U_{e,u_e} is a number, the equality always holds. □

COROLLARY 6.5. $\forall k \in \mathbb{N}$, $[\![\mathsf{CX}]\!]^{\sharp}$, $[\![\mathsf{Ry}_{k\pi}]\!]^{\sharp}$ $[\![\mathsf{Rx}_{k\pi}]\!]^{\sharp}$ $[\![\mathsf{Rz}_k]\!]^{\sharp}$, *are complete.*

We now focus on the encoding operators E, which, unlike the parametric operators U, depend on the environment and thus represent families of operators. This dependency prevents us from characterizing their completeness by restricting the operator's structure alone. Instead, we turn our attention to *local completeness*, which can be studied by restricting the class of possible inputs.

THEOREM 6.6. $[\![\mathsf{E}]\!]^{\sharp}$ *is (local) complete on* $\Sigma \in \wp(\Sigma_c)$ *and* $\psi \in \Psi_c$ *if* $\exists u \in \mathcal{B}_c$. $\psi(u) = 1 \wedge \forall e \neq u$. $\psi(e) = 0$, *and* $\gamma_{\mathrm{env}} \circ \alpha_{\mathrm{env}}(\Sigma) = \Sigma$ *namely, for any* x *the set* $\{ \sigma(x) \mid \sigma \in \Sigma \}$ *is a closed interval.*

PROOF. Let $\overline{\mathsf{S}} = \alpha_{\mathrm{env}}(\Sigma)$. As before, by simplifying Equation 5 and Definition 5.5, we can say that $[\![\mathsf{S}]\!]_{\overline{\mathsf{S}}}^{\sharp}(\alpha_{\mathrm{st}}(\psi))$ is complete if and only if $\forall e \in \mathcal{B}_c$. $\alpha_{\mathrm{CI}} \left(\{ (\!|\mathsf{E}_{e,u}|\!)_{\Sigma} \cdot \psi(u) \mid \sigma \in \Sigma \}\right) = (\!|\mathsf{E}_{e,u}|\!)_{\overline{\mathsf{S}}}^{\sharp} \cdot \alpha_{\mathrm{CI}}(\psi(u))$. Since $\psi(u) = 1$, the equation above holds if and only if $\alpha_{\mathrm{CI}} \left(\{ (\!|\mathsf{E}_{e,u}|\!)_{\Sigma} \mid \sigma \in \Sigma \}\right) = (\!|\mathsf{E}_{e,u}|\!)_{\overline{\mathsf{S}}}^{\sharp}$, i.e., $(\!|\mathsf{E}_{e,u}|\!)_{\overline{\mathsf{S}}}^{\sharp}$ is complete. From Subsection 5.3, we know that $(\!|\mathsf{E}_{e,u}|\!)_{\overline{\mathsf{S}}}^{\sharp}$ is defined as the *bca* of $(\!|\mathsf{E}_{e,u}|\!)_{\Sigma}$. Therefore, if $\gamma_{\mathrm{env}} \circ \alpha_{\mathrm{env}}(\Sigma) = \Sigma$, then $(\!|\mathsf{E}_{e,u}|\!)_{\overline{\mathsf{S}}}^{\sharp}$ is trivially complete [20, 32]. □

---

[8]When completeness is guaranteed only for specific inputs instead of universally, we refer to it as *local completeness* [11].
[9]A generalized permutation is a matrix with exactly one nonzero entry in each row and each column.

Finally, we consider when local completeness holds for the final measurement.

THEOREM 6.7. *If* $\forall e \in \mathcal{B}_c. \text{Re}(\Psi_c(e)) = 0 \vee \text{Im}(\Psi_c(e)) = 0$, $[\![M]\!]^\sharp$ *is (local) complete on* Ψ.

PROOF. By definition, if for all $\psi \in \Psi$ and for all bases e, $\text{Re}(\Psi_c(e)) = 0 \vee \text{Im}(\Psi_c(e)) = 0$, the measurement operators simply compute the square of a real numbers. The abstraction of the square function is always complete in real interval arithmetic by definition [51]. □

7 (ABSTRACT) VQC-BASED CLASSIFIER

Similarly to what happens when dealing with NNs, VQCs are primarily employed for data classification tasks [70]. In the previous section, we discussed how to abstract the execution of a VQC when its input is represented as an interval. In this section, we exploit both the concrete and abstract semantics of VQC for modeling the final step of its employment as a classifier: the classification function, i.e., the function deputed to map the output of the quantum computation to a corresponding class label.

7.1 VQC-based Classification

A VQC returns a probability distribution over the set of possible results. Crucially, for classification purposes, the output distribution of a VQC must be post-processed. As illustrated in the example in Section 3, classification is determined by the value of the first qubit. This means that, for instance, the probabilities of measuring 10 and 00 are aggregated into one class, while 11 and 01 are aggregated into another. More generally, the number of target classes is often smaller than the number of possible results, requiring a mapping from quantum outputs to class labels by summing the probabilities of grouped outcomes accordingly.

For a c $\in \mathcal{L}_{\text{VQC}}$, we introduce the function $\xi : \mathbb{A}_c \times \bigcup_{n \leq |QV_c|} QV_c^n \to \bigcup_{n \leq |QV_c|} \{0, 1\}^n$ that determines which class is the selected one, based on the outcome of the VQC execution. Let $\rho \in \mathbb{A}_c$ be the resulting distribution of a VQC, and let $\mathbf{q} \in QV_c^n$ be a tuple of n qubits, we define:

$$\xi(\rho, \mathbf{q}) \stackrel{\text{def}}{=} \arg \max_{b \in \{0,1\}^n} \sum_{e \in \mathcal{B}_c. e[\mathbf{q}]=b} \rho(e),$$

where $e[\mathbf{q}]$ denotes the projection of the bitstring e onto the positions in $\mathbf{q}$.

As an example, consider $\mathbf{q} = \langle q_0 \rangle$, meaning that we are interested in the classification task observing the value of the single ($n = 1$) qubit q_0. Let us consider the distribution resulting from the VQC in Section 3, i.e., $\rho = \{00 \mapsto 0.26, 01 \mapsto 0.21, 10 \mapsto 0.01, 11 \mapsto 0.52\}$, where the first bit corresponds to qubit q_1 and the second to q_0. In order to determine the selected class, we compute the aggregated probabilities: for $b = 0$, i.e., outcomes where $q_0 = 0$, the relevant bases are 00 and 10 ($00[\mathbf{q}] = 10[\mathbf{q}] = 0$), with $\rho(00) = 0.26$ and $\rho(10) = 0.01$, hence $\rho(10) + \rho(00) = 0.27$. Similarly, for $b = 1$, i.e., outcomes where $q_0 = 1$, the relevant bases are 01 and 11 ($01[\mathbf{q}] = 11[\mathbf{q}] = 1$), with $\rho(01) = 0.21$ and $\rho(11) = 0.52$, so $\rho(01) + \rho(11) = 0.73$. Since $0.73 > 0.27$, we have: $\xi(\rho, \langle q_0 \rangle) = 1$. That is, the most likely class according to the classification rule based on q_0 is the one associated with 1.

We can now define the semantics of the whole VQC classifier. Without loss of generality, we assume that the execution of a VQC always starts from a specific initial state where all quantum variables are in the value 0. Thus, given c $\in \mathcal{L}_{\text{VQC}}$ ($|QV_c| = n$), we define the initial state $\psi_0 \in \Psi_c$ as:

$$\psi_0 \stackrel{\text{def}}{=} \lambda e \in \mathcal{B}_c. \begin{cases} 1 & e = 0^n \\ 0 & \text{othewise} \end{cases} \tag{9}$$

If qubits $\mathbf{q} \in QV_c^n$ ($n \leq |QV_c|$) are the qubit encoding the classical dataset to be classified, the semantics of the whole VQC classification algorithm can be modeled as $\xi([\![c]\!]_\sigma(\psi_0), \mathbf{q})$. We abuse

notation by using the function ξ also to denote its additive lift to sets of distributions:

$$\xi(\llbracket c \rrbracket_\Sigma(\psi_0), \mathbf{q}) \stackrel{\text{def}}{=} \left\{ \xi(\llbracket c \rrbracket_\sigma(\psi_0), \mathbf{q}) \mid \sigma \in \Sigma \right\}. \tag{10}$$

An explicit definition of the classification functions is given in [60], where the authors define a classification function based on the measurement of a single qubit followed by classical post-processing with a tunable bias. This bias, implemented purely at the classical level by adjusting the decision threshold, allows the model to learn more flexible decision boundaries without increasing the quantum circuit depth. For the sake of simplicity but without loss of generality, we omit such bias terms in this paper. Nonetheless, our framework is general enough to accommodate them.

7.2 Abstract VQC-based classification

The classification process described above, which assigns a unique class based on the output distribution, applies to concrete executions of the VQC. However, when abstracting the semantics of the VQC, the output is no longer a single probability distribution but an abstract distribution, where each standard basis vector is associated with an interval of possible probabilities. As a consequence, the aggregated probabilities for different classes may result in overlapping intervals, making it impossible to determine a unique selected class with certainty.

A first step toward managing the uncertainty introduced by abstraction is to return a *set* of potential selected classes, i.e., all classes that could be chosen given the overlapping probability intervals. This allows us to assess whether all possible classifications are acceptable, or a refinement of the abstraction is necessary to reduce ambiguity, as it has been formalized for verifying abstract robustness in NN [31, 44].

Hence, we first abstract the ξ function in order to select only the intervals with a lower bound greater than any other interval distribution. Specifically, $\xi^\sharp : \varrho^\sharp \times \bigcup_{n \leq |QV_c|} QV_c^n \to \bigcup_{n \leq |QV_c|} \wp(\{0,1\}^n)$ selects the set of potential classes with higher probability. Formally, given $\mathbf{q} \in QV_c^n$

$$\xi^\sharp(\varrho^\sharp, \mathbf{q}) \stackrel{\text{def}}{=} \left\{ b \in \{0,1\}^n \;\middle|\; \nexists b' \in \{0,1\}^n. \sum_{e \in \mathcal{B}_c. e[\mathbf{q}]=b} \varrho^\sharp(e) <_{\mathbb{R}\mathbb{I}} \sum_{e \in \mathcal{B}_c. e[\mathbf{q}]=b'} \varrho^\sharp(e) \right\},$$

where the summation $\sum$ is computed according to the real interval arithmetic. Since we are working with abstract distributions, we may have an overlap of classes that can be selected. In fact, we note that the image of the function is in $\wp(\{0,1\}^n)$. To see this on an example, let us consider the following abstract distribution: $\varrho^\sharp = \{000 \to [0.2, 0.46], 001 \to [0, 0.2], 010 \to [0.1, 0.03], 101 \to [0.11, 0.6], 111 \to [0.1, 0.6]\}$ on 3 qubits $\langle q_2, q_1, q_0 \rangle$ standard basis vectors, where we omit the ones mapped to the singleton $[0, 0]$. Let $\mathbf{q} = \langle q_2, q_0 \rangle$, we first compute $\sum_{e \in \mathcal{B}_c. e[\mathbf{q}]=b} \varrho^\sharp(e)$ and we obtain: $[0.29, 0.56]$ for 00, $[0, 0.2]$ for 01, $[0.22, 1.2]$ for 11 and $[0, 0]$ for 10.[10] In this case, there is no single class whose lower bound exceeds the upper bounds of all other classes. However, we can observe that the configurations $\{01, 10\} \in \wp(\{0, 1\}^2)$ have definitively lower probabilities than $\{00, 11\} \in \wp(\{0, 1\}^2)$. Therefore, the function $\xi^\sharp$ will return the latter set of configurations, whose probability intervals overlap but are still greater than those of the others.

Hence, given $\mathbf{q} \in QV_c^n$ ($n \leq |QV_c|$) consisting in the list of qubits to consider for the classification, and the abstract initial state $\overline{V}_0 = \alpha_{st}(\{\psi_0\})$, the abstract semantics of the classification algorithm is:

$$\xi^\sharp(\llbracket c \rrbracket_{\overline{S}}^\sharp(\overline{V}_0), \mathbf{q})$$

whose task consists of computing such approximated classification, potentially returning a *set* of possible selected results. Then we can observe that abstract classification is also sound.

---

[10]For the sake of readability, we assume that the result represents a map from the qubits of interest to their values, i.e., if $\mathbf{q} = \langle q_2, q_0 \rangle$ then 01 stands for $q_2 \mapsto 0$ and $q_0 \mapsto 1$.

PROPOSITION 7.1. $\forall \Sigma \in \wp(\Sigma_c), \mathbf{q} \in QV_c^n \ (n \leq |QV_c|). \ \xi(\llbracket c \rrbracket_\Sigma(\psi_0), \mathbf{q}) \subseteq \xi^\sharp(\llbracket c \rrbracket_{\alpha_{env}(\Sigma)}^\sharp(\overline{V}_0), \mathbf{q})$, i.e., the abstract classification is sound.

7.3 Robustness Verification of (Abstract) VQC Classifier

In this section, we discuss the robustness verification problem, sometimes referred to as *stability*, of VQCs, i.e., the problem of verifying that perturbations of the input do not affect the output classification. The following formalization can be easily extended to the verification of other properties, such as safety, as in classical NN-Verification [40].

Specifically, we focus on input perturbations bounded by an ℓ_∞-norm ball centered at a nominal input $x \in \mathbb{R}^n$, formally defined as: $C_\infty(x, \varepsilon) = \{x' \in \mathbb{R}^n \mid \|x' - x\|_\infty \leq \epsilon\}$. In practice, we test the circuit on abstract input intervals in the form $\Sigma_{\sigma,\epsilon} \overset{def}{=} \{ \sigma' \mid \|\sigma(x) - \sigma'(x)\|_\infty \leq \epsilon \}$, where σ is the perturbated input (environment) and ϵ represents the noise applied to such input. We now formally define the robustness verification problem for VQC-based classifiers.

Definition 7.2 (Robustness Verification of VQC (RVVQC)). Let $c \in \mathcal{L}_{VQC}, \epsilon \in \mathbb{R}$ the input perturbation, and $\mathbf{q} \in CV_c^n \ (n \leq |QV_c|)$, the list of output observed qubits. We say that c is *robust*, w.r.t. $\mathbf{q}$ and ϵ, (denoted $c \models_{\mathbf{q},\epsilon} RVVQC$) iff $\forall \sigma \in \Sigma_c$

$$\xi(\llbracket c \rrbracket_{\Sigma_{\sigma,\epsilon}}(\psi_0), \mathbf{q}) \subseteq \{\xi(\llbracket c \rrbracket_\sigma(\psi_0), \mathbf{q})\} \tag{11}$$

This definition tells us that a VQC classifier for c is robust whenever the set of resulting classes obtained by executing the VQC classifier on all the considered perturbations of the input σ is the singleton set containing only the class obtained by executing the VQC classifier precisely on σ.

Now we have to understand the computability and complexity concerns of such VQC verification issues. We can observe that, in the RVVQC problem, the input set (of environments) has the same cardinality as reals, being any possible association between (a finite set of) variables and reals This makes, in general, the verification problem (but also the corresponding confutation problem) not computable. We can observe that this happens independently of the potential recursivity (i.e., decidability in finite time) of the property to verify on the single input. This means that, in order to computationally treat the verification problem, we need to restrict the input space to *computable real numbers with a fixed maximal precision* only (in practice, this is the set of floating-point numbers), namely those reals that can be computed, to within any desired precision, by a finite, terminating algorithm.[11] This makes the input set space a countable and discrete subset of reals [49] and Equation 11 decidable, being $\Sigma_{\sigma,\epsilon}$ discrete and finite. In the following part of the paper, we consider as input environments only those defined by mapping variables to computable and discrete reals.

PROPOSITION 7.3. *Let $c \in \mathcal{L}_{VQC}, \epsilon \in \mathbb{R}$, and $\mathbf{q} \in CV_c^n \ (n \leq |QV_c|)$*

(1) $CP \overset{def}{=} \{ c \mid c \not\models_{\mathbf{q},\epsilon} RVVQC \}$ *is computable (recursive enumerable).*

(2) $VP \overset{def}{=} \{ c \mid c \models_{\mathbf{q},\epsilon} RVVQC \}$ *is not computable;*

The intuition is that, for computing VP we have to execute the VQC on an infinite (even if computable) set of inputs, while for computing CP it is sufficient to find one input not satisfying Equation 11, and if it exists, we can find it in finite time over a discrete space of inputs (e.g., by using a technique called *dovetail* for searching on the input space without diverging). Trivially, if we restrict to a finite set of inputs, then both problems collapse to recursive problems. Nevertheless, even when made decidable, providing an answer to the RVVQC problem requires testing the VQC on a potentially huge set of inputs (depending on the chosen precision in approximating reals), which is clearly hard.

---

[11]A computable number [is] one for which there is a Turing machine which, given n on its initial tape, terminates with the nth digit of that number [encoded on its tape] [49].

THEOREM 7.4. *RVVQC is NP-hard.*

PROOF. To show the hardness of the problem, we show the following reduction 3-SAT $\preccurlyeq_k$ RVVQC. In particular, we will show that any 3-SAT instance on a Boolean formula Φ can be converted into a VQC c such that the property Π is satisfied on c if and only if Φ is satisfiable.

Let $\Phi = (C_1 \wedge \ldots \wedge C_m)$ be a 3-conjunctive normal form (CNF) over a set of Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$. Each formula is a conjunction of clauses, where each clause is a disjunction of exactly three literals. Each literal is either a variable x_k or its negation $\neg x_k$, for some $k \in \{1, \ldots, n\}$. An assignment $a : X \rightarrow \{0, 1\}$ satisfies Φ if it satisfies all the clauses simultaneously.

For the reduction, we first note that Φ can be viewed as a Boolean circuit which computes a function $f : a \rightarrow \{0, 1\}$, such that $f(a) = 1$ if and only if a satisfies Φ. Hence, from [37, Section 7] we can transform f into a quantum circuit $\mathsf{U}_{3-\text{SAT}}$ acting on poly(n) qubits. However, for the complete reduction, since we aim to obtain a VQC from Φ, we first need to introduce an *encoding gadget* E which takes as input a and encodes it for the $\mathsf{U}_{3-\text{SAT}}$. In Fig. 4 we report a schematic overview of E.
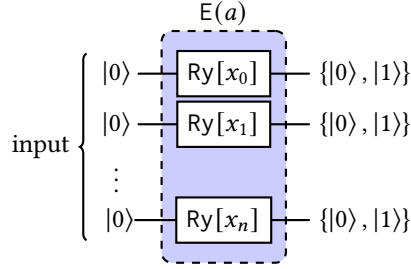

Fig. 4. *Encoding Gadget* E

In detail, the input of E corresponds to $|X|$ qubits initialized to $|0\rangle$. Then for a given an assignment a for $\{x_1 \rightarrow \{0, 1\}, \ldots, x_n \rightarrow \{0, 1\}\}$, we define a static environment σ_a as:

$$\sigma_a(x_i) \leftarrow \begin{cases} \pi & \text{if } a(x_i) = 1 \\ 0 & \text{if } a(x_i) = 0 \end{cases} \tag{12}$$

For the output of E, we then prove the following.

LEMMA 7.5. *For any assignment $a : X \rightarrow \{0, 1\}$, the evaluation of the encoding gadget E w.r.t. the static environment σ_a produces an output which is consistent with a.*

PROOF. The proof is straightforward by observing that if the original assignment for an x_i is 1, then the gate rotation $\mathsf{Ry}[\sigma_a(x_i) = \pi]$ will produce $|1\rangle$ as output. On the other hand, if $x_i = 0$ then $\mathsf{Ry}[\sigma_a(x_i) = 0]$ results in the identity leaving the state as $|0\rangle$.   □

By combining the *encoding gadget* E with the $\mathsf{U}_{3-\text{SAT}}$ circuit, we obtain a VQC c as desired. Clearly, the reduction from any 3-SAT instance Φ to the VQC c can be carried out in polynomial time. Let Φ have n variables and m clauses. The encoding gadget E introduces one R_y rotation gate per variable, thus requiring $O(n)$ gates. For each clause, the construction of the $\mathsf{U}_{3\text{-SAT}}$ circuit involves a constant number of controlled-NOT and negation gates (to implement 3-literal OR via De Morgan), requiring $O(1)$ gates per clause. The final conjunction over the clause outputs requires $m - 1$ additional controlled operations. Hence, the total number of quantum gates and qubits used in c is $O(n + m)$.

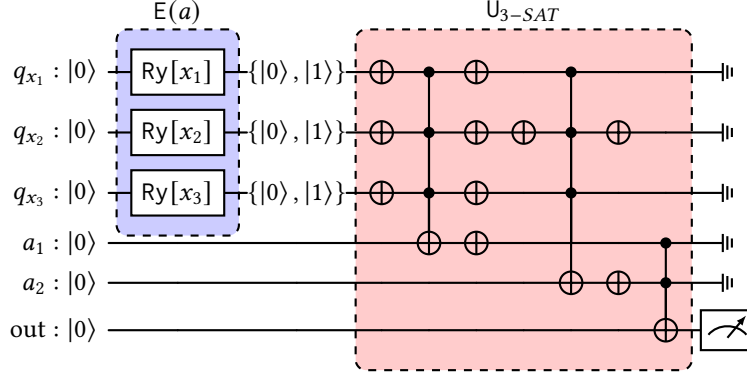To complete the proof, we need to show the following.

Fig. 5. Complete reduction of a formula $\Phi(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3)$ into a VQC c.

LEMMA 7.6. *Any 3-SAT formula Φ can be reduced into a VQC c and a property $\Pi = \{1\}$, such that Π is satisfiable on c if and only if Φ is satisfiable.*

PROOF. Without loss of generality, we will prove the lemma exploiting the example of the reduction of a simple formula Φ with $n = 3$ variables and $m = 2$ clauses into a VQC c reported in Fig. 5.

$\boxed{\Longrightarrow}$

We need to show that if $\xi([\![c]\!]_{\sigma_a}(\psi_0), \mathbf{q}) \subseteq \Pi$, i.e., the concrete execution of the VQC with the environment σ_a, then there exists an assignment a such that $\Phi(a) = 1$. Assume that $\xi([\![c]\!]_{\sigma_a}(\psi_0), \mathbf{q}) \subseteq \Pi = \{1\}$. This means that when the VQC c is executed and measured, the output qubit is observed in the state 1. By construction, c outputs $|1\rangle$ if and only if all clause qubits a_i (for $i = 1, \ldots, m$) are set to $|1\rangle$, meaning that each corresponding clause of the formula Φ is satisfied. Therefore, measuring the output qubit in state $|1\rangle$ implies that the assignment used by the circuit satisfies every clause in Φ. By Lemma 7.5, each assignment a to the Boolean variables of Φ corresponds to a valid and consistent quantum state prepared by the encoding gadget E. Hence, if the output qubit of c is measured as $|1\rangle$, then there must exist at least one assignment a such that $\Phi(a) = 1$, which means that Φ is satisfiable.

$\boxed{\Longleftarrow}$

Fix an assignment a for $\Phi(x_1, x_2, x_3)$, such that $\Phi(a) = 1$. The first step in the reduction is to produce the output of the *encoding gadget* E, for the assignment a, i.e., $E(a)$. From Lemma 7.5 we know that this gadget will produce an output consistent with a, for instance if the assignment a produce $\{x_1 \rightarrow 0, x_2 \rightarrow 1, x_3 \rightarrow 1\}$ then the corresponding output of $E(a)$ which becomes the input for U_{3-SAT} will be $\{|0\rangle, |1\rangle, |1\rangle\}$. The U_{3-SAT} circuit, first applies the negation gates ($\oplus$) to $q_{x_1}, q_{x_2}, q_{x_3}$ to get their negations. By applying a multi-controlled NOT, it computes $(\neg q_{x_1} \wedge \neg q_{x_2} \wedge \neg q_{x_3})$, storing the result in a_1. By negating the result of $a_1 = \neg(\neg q_{x_1} \wedge \neg q_{x_2} \wedge \neg q_{x_3}) = (q_{x_1} \vee q_{x_2} \vee q_{x_3})$ it encodes the OR operation for the first clause, as performed in Φ. Similarly, it is done for the second clause, so the circuit first restores the original values of $q_{x_1}, q_{x_2}, q_{x_3}$ by performing a negation on all the values. It then compute the negation only on q_{x_2}, storing the result in $a_2 = (x_1 \wedge \neg x_2 \wedge x_3)$. After this, it exploits again a multi-controlled NOT to get the result of $a_2 = \neg(x_1 \wedge \neg x_2 \wedge x_3) = (\neg x_1 \vee x_2 \vee \neg x_3)$. Finally, to encode the AND of the clauses, it simply performs the $\oplus$ operation on a_1 and a_2, storing the final result in $out = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3)$, which is precisely the original formula

Φ. Hence if a satisfies Φ then it also necessarily holds by construction that $a_1, a_2 = |1\rangle$, which once measured, will produce $1 \subseteq \Pi$.

□

The proof is complete.

□

To address this issue, we investigate whether our abstract interpretation framework can, in some sense, mitigate the inherent complexity of the problem. From Proposition 7.1, we know that classification based on the abstract semantics of a VQC is sound. This soundness ensures conservativeness with respect to the concrete computations—that is, no concrete behaviors are missed, and thus no possible classifications are lost. Hence, we can use the computed over-approximation in order to verify a desirable property RVVQC. Specifically, if the abstract output assigns a strictly higher probability to one class compared to all others, then—by the soundness of the abstraction—that class is guaranteed to be selected in the concrete execution as well.

PROPOSITION 7.7 (ABSTRACT RVVQC). *Consider a VQC c and $\mathbf{q} \in \mathrm{CV}_c^n$ ($n \leq |\mathrm{QV}_c|$). Given the initial abstract state $\overline{\mathrm{V}}_0$. Then c is robust if $\forall \sigma \in \overline{\Sigma}_c$*

$$\xi^{\sharp}(\llbracket c \rrbracket^{\sharp}_{\alpha_{\mathrm{env}(\Sigma_{\sigma,\epsilon})}}(\overline{\mathrm{V}}_0), \mathbf{q}) \subseteq \{\xi(\llbracket c \rrbracket_{\sigma}(\psi_0), \mathbf{q})\}.$$

However, due to the over-approximation introduced by abstraction, the abstract semantics may produce overlapping probability intervals across classes, making the classification result potentially uncertain and the verification outcome potentially inconclusive.

As a result, the potential overlap in the abstract output can distort the classification outcome, falsely declaring a VQC as potentially unsafe (i.e., the function cannot return a single class), even when it is actually safe, a phenomenon commonly referred to as a false alarm due to abstraction. Moreover, in general, from the computational complexity point of view, we cannot guarantee any improvement, as stated by the following result.

PROPOSITION 7.8. *Abstract RVVQC is NP-hard.*

Nonetheless, as in classical NN-Verification, abstract interpretation plays a crucial role in providing *provable* guarantees on the classifier. Indeed, we may reduce the number of executions to perform since each abstract execution captures several concrete executions, but by formally tackling the issue, we have an even more important gain. In fact, by using abstract interpretation, we can exploit the framework to *formally reason* about precision, the main source of uncertainty. In this context, precision is a fundamental challenge when dealing with approximation, and if we are able to identify the sources of imprecision, we may be able to potentially reduce it, reducing, as a consequence, uncertainty.

8 RECOVERING PRECISION IN ABSTRACT VQC VERIFICATION

As discussed in the previous section, the interval-based abstract semantics is sound but incomplete. In this section, we introduce some techniques to mitigate the resulting loss of precision.

Clipping the Intervals. In Example 6.3, we observe that the abstraction introduces components with norms greater than one. The problem is in the imaginary part of the amplitude associated with the basis 1 exceeding 1, due to the over-approximation error. To mitigate this issue, a possible solution is to remove such invalid elements by clipping all intervals to lie within $[-1, 1]$. That is, at each step of the computation, we intersect each component of the abstract state with the interval $[-1, 1]$. Formally, we define a function $\mathrm{clip}_{\mathrm{CI}} : \overline{\mathbb{V}}_c \rightarrow \overline{\mathbb{V}}_c$ as
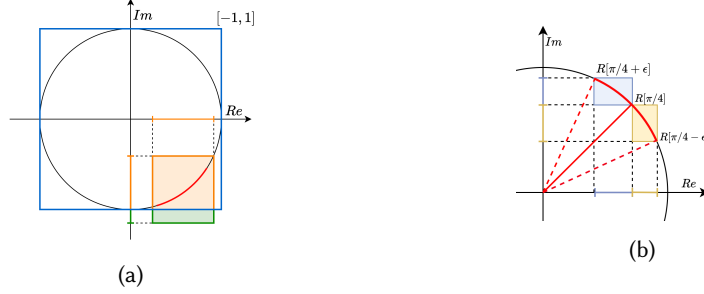
Fig. 6. An example of the clipping function (a), and of the refinement based on splitting intervals (b)

$\text{clip}_{\text{CI}} \stackrel{\text{def}}{=} \lambda \overline{V}. \left(\lambda e \in \mathcal{B}_c.\langle \text{Re}(\overline{V}(e)) \cap [-1, 1], \text{Im}(\overline{V}(e)) \cap [-1, 1] \rangle \right)$. A 2D illustration is shown in Fig. 6a.

PROPOSITION 8.1. *The clip operation is sound for all quantum statements, i.e.,* $\forall \Sigma \in \wp(\overline{\Sigma}_c)$, $\alpha_{\text{CI}}(\llbracket s \rrbracket_{\Sigma}(\psi_0)) \dot{\leq}_{\text{CI}} clip_{\text{CI}}(\llbracket s \rrbracket^{\sharp}_{\alpha_{\text{CI}}(\Sigma)}(\overline{V}_0))$

PROOF. The soundness follows from that given a $\psi \in \llbracket s \rrbracket_{\Sigma}(\psi_0)$, $\forall e \in \mathcal{B}_c$, $-1 \leq \text{Re}(\psi(e)) \geq 1$ and $-1 \leq \text{Im}(\psi(e)) \geq 1$. Thus the elements eliminated by the clip_{CI} are not in $\alpha_{\text{CI}}(\llbracket s \rrbracket_{\Sigma}(\psi_0))$. □

Similarly, after performing a measurement, the resulting probability intervals may have upper bounds exceeding 1. Since measurement outcomes must lie within the range $[0, 1]$, we can define another function that applies a clipping by intersecting the interval with $[0, 1]$.
Formally, we can define a function $\text{clip}_{\text{RI}} : \overline{\mathbb{D}}_c \rightarrow \overline{\mathbb{D}}_c$ as $\text{clip}_{\text{RI}} \stackrel{\text{def}}{=} \lambda \varrho^{\sharp}. \left(\lambda e \in \mathcal{B}_c.\varrho^{\sharp}(e) \cap [0, 1] \right)$. For similar consideration as in Proposition 8.1, we have

PROPOSITION 8.2. *The clip operation on the measurement is sound.*

Since both clip functions are sound, we can guarantee that applying the clip operation at each step of the abstract semantics is sound.

'Symbolic' Execution of Parametric Circuit. Another technique employed in classical NN-Verification for reducing the over-approximation is called *symbolic interval propagation* [67]. At a high level, the idea is to track dependencies between variables symbolically, during abstract computation, reducing over-approximation compared to naive interval propagation methods. However, its precision drops in the presence of non-convex transformations, and it often requires interval concretization to remain tractable.

Inspired by this approach, we investigate whether this simple yet effective technique can also be applied in a quantum setting. We begin by observing that, due to the linearity of the parametric part of the VQC, the evolution of a concrete quantum state follows the standard rule of operator composition, i.e., matrix multiplication. Specifically, given a vector state $|\psi\rangle$ and two unitary operators U and U', $U' \cdot (U \cdot |\psi\rangle) = (U' \cdot U) \cdot |\psi\rangle$. On the other hand, when working with abstract states, given an abstract state $\overline{V}$ and two operations U and U', the abstract semantics satisfy: $\llbracket U' \cdot U \rrbracket^{\sharp}(\overline{V}), \dot{\leq}_{\text{CI}} \llbracket U' \rrbracket^{\sharp} \circ \llbracket U \rrbracket^{\sharp}(\overline{V})$. This inclusion reflects the fact that by composing U and U' first, we can perform the concrete operations exactly (without over-approximation), and only afterwards apply a single abstract operation to the result, thus reducing the accumulation of over-approximation errors in the analysis.

Example 8.3. In order to understand the idea in practice, consider the example in Subsection 5.4. In this case, if we first compute the full operator $O \stackrel{\text{def}}{=} (\text{Ry}^{q_1}_{-0.69} \cdot \text{Ry}^{q_0}_{3.27}) \cdot \text{CX}^{q_0, q_1} \cdot \text{Ry}^{q_1}_{0.99} \cdot \text{Ry}^{q_0}_{-0.50}$ and

then apply it to the initial abstract state, we can directly compute the final distribution as:

$$\varrho_s^\sharp = [\![M]\!]^\sharp \circ [\![O]\!]_{\overline{S}}^\sharp(\overline{V}_1) = \begin{Bmatrix} 00 \mapsto [0.165, 0.394] & 01 \mapsto [0.128, 0.410] \\ 10 \mapsto [0, 0.068] & 11 \mapsto [0.320, 0.698] \end{Bmatrix}. \tag{13}$$

Thus, we find q_0 equals to 0 w.p. $[0.165, 0.462]$ and equals to 1 w.p. $[0.448, 1.108]$. We clearly observe that the intervals are narrower compared to those in $\varrho^\sharp$ from Equation 8, which was obtained by applying the operators step by step. However, we are currently unable to verify that the VQC is robust with respect to the initial input $x = [6, 2.7]$ perturbed by an $\epsilon = 0.5$. We can improve the abstraction process by introducing another technique, called iterative refinement.

Iterative Refiment. In classical NNs, one common technique to improve the precision of interval analysis is to recursively split the input interval into smaller sub-intervals [67]. For instance, analyzing the network over a broad input range such as $[0, 10]$ might produce a loose output bound like $[2, 30]$. However, if we divide the input into two smaller intervals, say $[0, 5]$ and $[5, 10]$, and analyze each one separately, we might obtain tighter bounds such as $[2, 12]$ and $[10, 20]$, respectively. Taking the union of these results yields a refined output bound of $[2, 20]$, improving the overall precision of the analysis. This strategy helps reduce overestimation caused by input dependencies. Its effectiveness is supported by the fact that NNs are Lipschitz continuous, which ensures that repeated input refinement will eventually converge to arbitrarily tight output bounds within a finite number of steps.

In our setting, the encoding functions used to generate quantum states, such as sine and cosine, are continuous. This continuity allows us to adopt a similar strategy; by partitioning the input domain and analyzing smaller subregions independently, we can derive more precise output intervals. In Fig. 6b, we provide a 2D visualization of this idea. Instead of computing the abstract rotation $R[\pi/4 - \epsilon, \pi/4 + \epsilon]$ all at once, we split the input interval and compute separately $R[\pi/4 - \epsilon, \pi/4]$ and $R[\pi/4, \pi/4 + \epsilon]$. As shown in the figure, the union of these two results yields a tighter over-approximation than applying the rotation over the entire interval at once.

Once again, to provide a practical example of this technique, let us consider again the VQC c used in Subsection 5.4. To further improve precision, we can refine the input domain by splitting one of the two input intervals, say, x_0. We thus define two sub-environments: $\overline{S}_0: \{x_0 \mapsto [5.5, 6], x_1 \mapsto [2.2, 3.2]\}$, $\overline{S}_1: \{x_0 \mapsto [6, 6.5], x_1 \mapsto [2.2, 3.2]\}$, such that $\overline{S} = \overline{S}_0 \cup \overline{S}_1$. Executing the abstract semantics on these two environments, using symbolic execution, we obtain:

$$[\![c]\!]_{\overline{S}_0}^\sharp = \begin{Bmatrix} 00 \mapsto [0.176, 0.387] & 01 \mapsto [0.139, 0.355] \\ 10 \mapsto [0, 0.057] & 11 \mapsto [0.322, 0.674] \end{Bmatrix}, \; [\![c]\!]_{\overline{S}_1}^\sharp = \begin{Bmatrix} 00 \mapsto [0.195, 0.285] & 01 \mapsto [0.148, 0.367] \\ 10 \mapsto [0, 0.048] & 11 \mapsto [0.40, 0.635] \end{Bmatrix}.$$

Taking the union, we obtain:

$$[\![c]\!]_{\overline{S}_0}^\sharp \cup [\![c]\!]_{\overline{S}_1}^\sharp = \begin{Bmatrix} 00 \mapsto [0.176, 0.387] & 01 \mapsto [0.139, 0.367] \\ 10 \mapsto [0, 0.057] & 11 \mapsto [0.322, 0.674] \end{Bmatrix}.$$

This yields a more precise output compared to Equation 13. If we check the results of the VQC, the probability associated with measuring q_0 equals 0 is $[0.176, 0.444]$ while we have q_0 equal to 1 with a probability in $[0.461, 1.04]$, 1. Refining the abstraction confirms that the VQC in Fig. 2 is robust when the initial input $x = [6, 2.7]$ is perturbed by an $\epsilon = 0.5$.

Why not a 'Symbolic Encoding'? This raises the question of whether performing a symbolic execution of the encoding part of the circuit could lead to improved precision. As we will show below, unfortunately, the answer is no.

PROPOSITION 8.4. *The symbolic execution of the encoding part of a VQC does not improve precision.*

Proof. To prove the proposition, we show a counterexample. Let us consider the program $s := \mathsf{Rx}^q[x_0]; \mathsf{Ry}^q[x_1]$. We recall that the Ry is defined as: $\mathsf{Ry} = \begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}$ and the Rx definition is given in Equation 1. The small program s models a circuit in which we are encoding multiple classical data in the same qubit. Let us consider $x = [\pi/2 - \epsilon, \pi/2 + \epsilon]$ and $y = [\pi/3 - \epsilon, \pi/3 + \epsilon]$, where $\epsilon = 1$. First, we can execute the circuit naively, starting from the abstract state $\overline{\mathsf{V}}_0 = \{0 \mapsto \langle [1,1][0,0] \rangle, 1 \mapsto \langle [0,0],[0,0] \rangle \}$. In particular, $\overline{\mathsf{V}}_1 = [\![\mathsf{Ry}[y]^q]\!]_\mathsf{S}^\sharp [\![\mathsf{Rx}^q[x]]\!]_\mathsf{S}^\sharp (\overline{\mathsf{V}}_0) = \{0 \mapsto \langle [0.564, 0.66] + [0.306, 0.402] \rangle, 1 \mapsto \langle [0.306, 0.402] + [-0.66, -0.564] \rangle \}$.

We now execute the circuit keeping x and y symbolic. First we compute $[\![\mathsf{Rx}^q[x]]\!]_\mathsf{S}^\sharp (\overline{\mathsf{V}}_0)$, that results in the symbolic state: $\overline{\mathsf{T}}_1[x] = \begin{cases} 0 \mapsto \cos(x/2) \\ 1 \mapsto -\iota \cdot \sin(x/2) \end{cases}$. Then we compute symbolically $[\![\mathsf{Ry}[y]^q]\!]_\mathsf{S}^\sharp (\overline{\mathsf{T}}_1[x]) = \begin{bmatrix} \cos(y/2) \cdot \cos(x/2) + (-\sin(y/2)) \cdot (-\iota \sin(x/2)) \\ \sin(y/2) \cdot \cos(x/2) + \cos(y/2) \cdot (-\iota \sin(x/2)) \end{bmatrix}$. By substituting the values of x and y into the resulting state, we obtain $\overline{\mathsf{V}}_1$. Alternatively, we can use trigonometric formulas to manipulate the state, obtaining: $\overline{\mathsf{T}}_2[x,y] = \begin{cases} 0 \mapsto \langle [\cos(\frac{x+y}{2}) + \cos(\frac{x-y}{2})/2], [\cos(\frac{x-y}{2}) - \cos(\frac{x+y}{2})/2] \rangle \\ 1 \mapsto \langle [\sin(\frac{x+y}{2}) - \sin(\frac{x-y}{2})/2], [-\sin(\frac{x-y}{2}) - \sin(\frac{x+y}{2})/2] \rangle \end{cases}$. Now substituting the x and the y with the proper intervals, we have: $\overline{\mathsf{T}} = \{0 \mapsto \langle [0.548, 0.670][0.291, 0.413] \rangle, 1 \mapsto \langle [0.290, 0.413][-0.670, -0.548] \rangle \}$. that is noisier than $\overline{\mathsf{V}}_1$. $\square$

We have shown, with a simple example, that the symbolic encoding does not always improve accuracy. This is due to the fact that trigonometric substitutions do not always improve precision.

9 EVALUATION

In this section, we evaluate our abstract interpretation-based framework for the robustness verification of VQC classifiers. In particular, we start by considering two standard dataset benchmarks in the classification literature, namely Iris [55] and MNIST [39]. To keep the paper self-contained, we briefly summarize the main characteristics of these datasets here. The Iris dataset consists of 150 samples of iris flowers classified into three species, such as *Iris setosa*, *Iris virginica*, and *Iris versicolor*. The input to the classifier consists of four features, such as the length and width of the sepals and petals, all in centimeters, respectively. In our setting, we only consider a binary classification task between *Iris setosa* and *Iris versicolor*. The MNIST dataset of handwritten digits, containing 60,000 training and 10,000 testing examples, each represented as a grayscale image of size 28×28 pixels. In our setting, we consider the binary classification of downscaled to 4×4 image digits 0 and 1 (denoted MNIST[0,1]), and digits 2 and 6 (denoted MNIST[2,6]).

Although verification typically assumes a pre-trained model, in order to assess how different levels of accuracy affect the robustness of the VQC, we train and verify three distinct models using our proposed framework as VQC-based classifiers. Specifically:

- **QCL** [50] is a representative hybrid classical-quantum framework that employs a VQC composed of a nonlinear quantum encoding circuit for input data and a low-depth variational quantum circuit. We train and verify this model on Iris. The circuit is represented in Fig. 7.
- **CCQC** [60] is a low-depth hybrid VQC framework designed for supervised learning. See Fig. 8 for the circuit representation. Unlike the previous approach, classification is performed based on the measurement outcomes combined with a trainable bias term. We train and verify this model on the Iris dataset. The circuit is represented in Fig. 8.
- The final models [36], which we refer to as **PV**, are a variation of the previous approaches and employ angle encoding [53] and a parametric circuit to mitigate flat cost function landscapes during optimization [33]. We train this model on MNIST. The circuit is represented in Fig. 9.

Fig. 7. The QCL model, used to classify a 4-feature input data.

Fig. 8. The CCQC model used on the iris dataset. Here $R3_{w_i, w_j, w_k} = Rz_{w_k} \cdot Ry_{w_j} \cdot Rx_{w_i}$

Fig. 9. The PV model, used to classify a 16-feature input data.

Training Phase. As described in Sec. 3, a VQC, like a deep NN, consists of a parameterized quantum circuit whose parameters are optimized during training to minimize a cost function, typically defined as the margin between the predicted and target classes [8]. For clarity, the left part of Fig. 10 presents a simplified version of the training algorithm used for the QCL model on the Iris dataset, while on the right side of the figure, we provide an illustrative view of the decision boundaries for the CCQC model on the same dataset. Specifically, we adopt a supervised training procedure for each VQC tested, exploiting the Pennylane framework [7]. For each training example, the prediction is computed via a VQC, and the parameters θ are updated using gradient-based optimization. This process is repeated over multiple epochs to improve classification performance. While the pseudocode illustrates a generic Stochastic Gradient Descent (SGD) update, in our experiments, we employ different optimization strategies, such as PennyLane's *NesterovMomentumOptimizer* [7], using the quantum state encoding defined in [52]. The right table in Fig. 11 shows the test set accuracy for each model tested on its corresponding dataset.

Verification Phase. After the training phase, we perform a formal verification step to assess the robustness of trained VQCs to input perturbations. To this end, we extend the Pennylane framework [7] to support our abstract interpretation-based formulation, which operates over intervals of real and complex numbers. This extension enables reachability analysis through the circuit, allowing us to reason about how input uncertainty propagates through quantum operations.

The verification process is summarized on the left side in Fig. 11. In detail, we show the algorithm used to compute the maximum perturbation ϵ each model can tolerate for a given input x and target class y. The VERIFY function is the core of the pipeline. It takes as input a trained VQC (i.e., with fixed parameters θ), a test input x, a class label y, and a perturbation radius ϵ. It encodes the input as intervals, i.e., each feature of x is extended to $[x_i - \epsilon, x_i + \epsilon]$, and propagates these intervals through the encoding and a symbolic representation of the variational layers of the circuit.

During interval propagation, both real and complex-valued amplitudes are tracked using interval arithmetic. The prediction is derived by identifying the output class whose associated probability interval has a lower bound that exceeds the upper bounds of all others, ensuring the prediction

SGD for QCL Binary Classification

1: **Input:** QCL with parameter θ, Training data $\{(X_i, Y_i)\}_{i=1}^{N}$, learning rate η, epochs E
2: **Initialize:** Random parameters θ
3: **for** epoch = 1 to E **do**
4:     **for** each (X_i, Y_i) in training set **do**
5:         $\hat{y}_i \leftarrow \text{QCL}(X_i, \theta)$
6:         $loss \leftarrow (\hat{y}_i - Y_i)^2$
7:         $g \leftarrow \nabla_\theta \ell_i$ ▷ Compute gradient
8:         $\theta \leftarrow \theta - \eta \cdot g$ ▷ Update parameters
9:     **end for**
10:    $a \leftarrow \text{ACCURACY}(\text{training set}, QCL, \theta)$
11: **end for**
12: **return** trained parameters θ
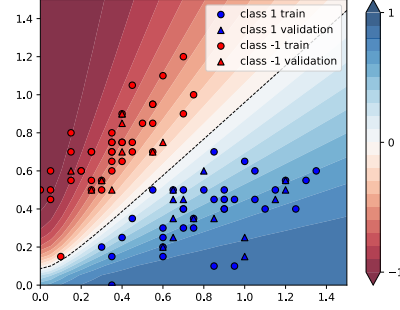
Fig. 10. Left: Training algorithm for a Variational Quantum Classifier using SGD. Right: Empirical decision boundary on the Iris dataset using CCQC classifier (100% accuracy).

is unambiguous under the entire input region. If this condition is not met, for instance if the intervals of two or more output classes overlap, an iterative refinement process is employed: the input interval with the largest uncertainty (i.e., the largest width) is split, and the verification is recursively applied to the sub-regions until a conclusive result is reached (i.e., all the subregions are safe or a single counterexample is discovered) or a specified precision is met.[12] Broadly speaking, the algorithm combines an exponential and binary search. It starts from a minimal $\epsilon_{\min}$ and iteratively doubles it until the robustness condition is violated or $\epsilon_{\max}$ is reached. Then, a binary search is used to converge to the largest ϵ that still guarantees robustness, with a desired precision τ.

Compute max ϵ perturbation

1: **Input:** VQC with trained parameter θ, input x, y class, $\epsilon_{\min}$, $\epsilon_{\max}$, τ tolerance threshold
2: $\epsilon \leftarrow \epsilon_{\min}$
3: **while** $\epsilon \leq \epsilon_{\max}$ **and** $\text{VERIFY}(VQC, x, \epsilon, y)$ == robust **do**
4:     $\epsilon \leftarrow 2 \cdot \epsilon$
5: **end while**
6: $high \leftarrow \min(\epsilon, \epsilon_{\max})$
7: $low \leftarrow \epsilon/2$
8: $max\_epsilon \leftarrow low$
9: **while** $high - low > \tau$ **do**
10:    $mid \leftarrow (low + high)/2$
11:    **if** $\text{VERIFY}(VQC, x, \epsilon, y)$ == robust **then**
12:       $max\_epsilon \leftarrow mid$
13:       $low \leftarrow mid$
14:    **else**
15:       $high \leftarrow mid$
16:    **end if**
17: **end while**
18: **return** $max\_epsilon$

Robustness Verification results

Model-Dataset	Accuracy (%)	Mean of max ϵ pert.
QCL-Iris	76%	0.07496±0.05
CCQC-Iris	100%	0.1244±0.04
PV-MNIST[0,1]	95%	0.0048±0.002
PV-MNIST[2,6]	78%	0.0022±0.001

Fig. 11. Left: the verification pipeline. Right: robustness verification results. The mean of the maximum ϵ is computed over 10 randomly selected inputs from the test set and the model with the highest accuracy during training.

---

[12]We also experimented with different heuristics for selecting the node to split during refinement—for example, choosing nodes at random—but the strategy described above consistently yielded the best results in terms of precision and was therefore used to generate the reported outcomes.

The right side of Fig. 11 reports the verification results on different quantum classifiers and datasets. The results reveal several interesting insights: `QCL-Iris` achieves an accuracy of 76% with a mean maximum ϵ of 0.07496. This relatively high perturbation tolerance reflects the low dimensionality and structured nature of the Iris dataset. On the other hand, `CCQC-Iris`, which adopts a different encoding or circuit structure, achieves perfect accuracy (100%) and significantly higher robustness (mean ϵ = 0.1244). This indicates better alignment between circuit expressivity and the problem structure, possibly due to improved use of parameterized gates. For the more complex MNIST dataset, particularly in binary classification tasks over digits [0,1] and [2,6], we observe a clear drop in robustness: 0.0048 and 0.0022, respectively. While `PV-MNIST[0,1]` still maintains high accuracy (95%), the low maximum ϵ highlights how small input perturbations can significantly alter the classification outcome in high-dimensional settings. This is even more evident for `PV-MNIST[2,6]`, where both accuracy and robustness are lower, suggesting that distinguishing these digits is harder for the model.

Overall, the results validate the effectiveness of our reachability-based verification approach in quantifying model robustness in quantum machine learning. The method not only provides formal guarantees but also reveals the limitations of current quantum models in handling high-dimensional, less separable data. These insights could drive the design of more robust circuits and encoding schemes in future work.

10 RELATED WORK

NN verification. Classical NN formal verification tools [40, 69], such as those developed in [30] and [64], adopt abstract interpretation techniques to conservatively approximate the network's behavior by propagating abstract domains (like intervals or zonotopes) through its layers. As demonstrated in this work, this approach allows efficient and sound verification of properties—such as robustness—by computing conservative bounds on the output set. A different category of tools, including CROWN, α-CROWN, and β-CROWN [68, 71, 73], employs linear relaxation methods to approximate non-linear activations with tight linear bounds, offering a balance between scalability and precision. These techniques are often integrated with Branch-and-Bound frameworks [13], such as MN-BaB [28], which partition the verification task into smaller subproblems—either by splitting the input perturbation space [67] or by dividing ReLU activations into linear segments [12, 13]. This combination enhances completeness and precision, though at the cost of increased computational complexity. More recently, emerging approaches have extended the scope of verification to identify all input regions that satisfy a given output specification [24, 38, 42, 43, 74].

VQC Robustness Evaluation. Robustness evaluation of variational quantum circuits remains largely unexplored. However, recent works have started to address this challenge. For example, QuanTest [63] proposes an adversarial testing framework that generates inputs maximizing a quantum entanglement adequacy criterion while exposing erroneous behaviors, using a gradient-based joint optimization approach. Other studies [41, 70] have investigated the vulnerability of VQC models to adversarial attacks, demonstrating that quantum classifiers, like their classical counterparts, can be misled by imperceptible perturbations on both classical and quantum inputs across a variety of tasks. Nonetheless, none of these works provides a framework for establishing provable robustness guarantees for a given VQC model. To the best of our knowledge, we are the first to introduce this important capability to the quantum machine learning literature.

Abstract Interpretation in Quantum Computing. Yu et al. [72] use abstract interpretation to reduce quantum state dimensionality by partitioning an n-qubit state into m lower-dimensional abstractions over $k < n$ qubits, enabling assertion verification in static circuits. Feng et al. [26] connect quantum Hoare logic and quantum incorrectness logic to abstract interpretation, showing

how each framework can derive the others. Abstract interpretation has also been used to analyze entanglement [4, 5, 35, 56], by defining sound abstract domains and semantics for quantum programs. Bichsel et al. [9] propose an abstract stabilizer simulator that efficiently overapproximates the effect of Clifford and non-Clifford gates, as well as measurements. The abstract domain uses complex intervals: quantum states are represented as sums of Pauli operators with complex coefficients, which are abstracted using interval arithmetic and abstract Pauli operators.

11 CONCLUSION

In this work, we have presented a novel framework for the formal verification of variational quantum circuits, which is based on the abstract interpretation theory. Our approach is grounded in interval abstractions, which, despite their non-relational nature and the resulting over-approximations, still enable effective verification, as demonstrated by our empirical evaluation. To enhance precision, we have introduced several strategies inspired by techniques developed for classical deep neural network verification.

Future Directions. We have observed that classification based on the abstract execution of a VQC may suffer from overestimation errors, preventing the system from producing a concrete output. However, recent work on classical NNs [31, 44] has shown that it is possible to relax the requirement of obtaining a precise answer by instead providing an abstract robustness guarantee. This involves applying abstraction not only to the input but also to the output, allowing for different degrees of precision. The verification task then becomes a question of whether the desired property holds within a specified level of abstraction, an idea closely related to Adequacy in static analysis [45]. We argue that this tunable approach to verification is promising and deserves further investigation.

Another promising direction for future work is to explore alternative techniques—beyond those discussed in Section 8—for recovering precision. In particular, methods such as circuit rewriting and simplification, including the use of ZX-calculus [17, 66], offer a compelling avenue. These techniques could be applied as a pre-processing step to simplify quantum circuits, for instance, by reducing the number of phase gates in the encoding, thereby potentially enhancing the overall precision of the analysis. The key challenge lies in understanding how to effectively integrate this powerful formalism into our verification framework and adapt it to our specific context.

Alternative representations of quantum states also offer valuable opportunities. One possibility could be the path-sum formalisms [1–3, 15, 59], which provide symbolic descriptions of quantum circuits that can be more concise than vector-based representations, and have been shown effective for verifying large Clifford+T circuits. Similarly, bit-wise simulation techniques [16, 22] allow memory-efficient encoding of quantum states, especially when entanglement is limited. We also plan to explore the use of the Ellipsoid domain [21, 27] to capture second-order relations between the amplitudes of a quantum state.

Finally, we plan to extend our framework to support the analysis of hybrid quantum-classical architectures [34, 47, 48], in which VQCs appear as components within larger machine learning pipelines. This includes exploring how input abstractions can propagate through both quantum and classical layers, as well as how robustness guarantees can be preserved end-to-end.

REFERENCES

[1] Matthew Amy. 2018. Towards Large-Scale Functional Verification of Universal Quantum Circuits. In *Proceedings of the 15th International Conference on Quantum Physics and Logic (QPL 2018) (EPTCS, Vol. 287)*. https://doi.org/10.4204/EPTCS.287.1

[2] Matthew Amy. 2019. *Formal Methods in Quantum Circuit Design.* Ph.D. Dissertation.

[3] Matthew Amy and Joseph Lunderville. 2025. Linear and Non-linear Relational Analyses for Quantum Program Optimization. *Proceedings of the ACM on Programming Languages* 9, POPL (Jan. 2025), 1072–1103. https://doi.org/10.

1145/3704873

[4]   Nicola Assolini, Alessandra Di Pierro, and Isabella Mastroeni. 2024. Abstracting Entanglement. In *Proceedings of the 10th ACM SIGPLAN International Workshop on Numerical and Symbolic Abstract Domains, NSAD 2024, Pasadena, CA, USA, 22 October 2024*, Vincenzo Arceri and Michele Pasqua (Eds.). ACM, 34–41. https://doi.org/10.1145/3689609.3689998

[5]   Nicola Assolini, Alessandra Di Pierro, and Isabella Mastroeni. 2025. A Static Analysis of Entanglement. In *Verification, Model Checking, and Abstract Interpretation - 26th International Conference, VMCAI 2025, Denver, CO, USA, January 20-21, 2025, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 15530)*, Shankaranarayanan Krishna, Sriram Sankaranarayanan, and Ashutosh Trivedi (Eds.). Springer, 50–71. https://doi.org/10.1007/978-3-031-82703-7_3

[6]   Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. 1995. Elementary gates for quantum computation. *Phys. Rev. A* 52 (Nov 1995), 3457–3467. Issue 5. https://doi.org/10.1103/PhysRevA.52.3457

[7]   Ville Bergholm, Josh A. Izaac, Maria Schuld, Christian Gogolin, and Nathan Killoran. 2018. PennyLane: Automatic differentiation of hybrid quantum-classical computations. *CoRR* abs/1811.04968 (2018). arXiv:1811.04968 http://arxiv.org/abs/1811.04968

[8]   Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. 2017. Quantum machine learning. *Nature* 549, 7671 (2017), 195–202.

[9]   Benjamin Bichsel, Anouk Paradis, Maximilian Baader, and Martin T. Vechev. 2023. Abstraqt: Analysis of Quantum Circuits via Abstract Stabilizer Simulation. *Quantum* 7 (2023), 1185. https://doi.org/10.22331/Q-2023-11-20-1185

[10] Roberto Bruni, Roberto Giacobazzi, Roberta Gori, and Francesco Ranzato. 2021. A Logic for Locally Complete Abstract Interpretations. *Proceedings - Symposium on Logic in Computer Science* 2021-June (2021). https://doi.org/10.1109/LICS52264.2021.9470608

[11] Roberto Bruni, Roberto Giacobazzi, Roberta Gori, and Francesco Ranzato. 2023. A Correctness and Incorrectness Program Logic. *J. ACM* 70, 2, Article 15 (March 2023), 45 pages. https://doi.org/10.1145/3582267

[12] Rudy Bunel, Jingyue Lu, Ilker Turkaslan, Philip HS Torr, Pushmeet Kohli, and M Pawan Kumar. 2020. Branch and bound for piecewise linear neural network verification. *Journal of Machine Learning Research* 21, 42 (2020), 1–39.

[13] Rudy R Bunel, Ilker Turkaslan, Philip Torr, Pushmeet Kohli, and Pawan K Mudigonda. 2018. A unified view of piecewise linear neural network verification. *Advances in Neural Information Processing Systems* 31 (2018).

[14] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. 2021. Variational quantum algorithms. *Nature Reviews Physics* 3, 9 (2021), 625–644.

[15] Clément Chareton, Sébastien Bardin, François Bobot, Vincent Perrelle, and Benoît Valiron. 2021. An Automated Deductive Verification Framework for Circuit-building Quantum Programs. In *Programming Languages and Systems. ESOP 2021 (Lecture Notes in Computer Science, Vol. 12648)*, Nobuko Yoshida (Ed.). Springer, Cham, 156–182. https://doi.org/10.1007/978-3-030-72019-3_6

[16] Yanbin Chen and Yannick Stade. 2023. Quantum Constant Propagation. In *Static Analysis - 30th International Symposium, SAS 2023, Cascais, Portugal, October 22-24, 2023, Proceedings (Lecture Notes in Computer Science, Vol. 14284)*, Manuel V. Hermenegildo and José F. Morales (Eds.). Springer, 164–189. https://doi.org/10.1007/978-3-031-44245-2_9

[17] Bob Coecke and Ross Duncan. 2009. Interacting Quantum Observables: Categorical Algebra and Diagrammatics. *CoRR* abs/0906.4725 (2009). arXiv:0906.4725 http://arxiv.org/abs/0906.4725

[18] Patrick Cousot. 2021. Dynamic interval analysis by abstract interpretation. *Formal Methods in Outer Space* 13065 (2021), 61–86. https://doi.org/10.1007/978-3-030-87348-6_4 Series Title: Lecture Notes in Computer Science.

[19] Patrick Cousot and Radhia Cousot. 1977. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages* (Los Angeles, California) *(POPL '77)*. Association for Computing Machinery, New York, NY, USA, 238–252. https://doi.org/10.1145/512950.512973

[20] Patrick Cousot and Radhia Cousot. 1979. Systematic design of program analysis frameworks. In *Proceedings of the 6th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages* (San Antonio, Texas) *(POPL '79)*. Association for Computing Machinery, New York, NY, USA, 269–282. https://doi.org/10.1145/567752.567778

[21] Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, David Monniaux, and Xavier Rival. 2006. Combination of Abstractions in the ASTRÉE Static Analyzer. In *Advances in Computer Science - ASIAN 2006. Secure Software and Related Issues, 11th Asian Computing Science Conference, Tokyo, Japan, December 6-8, 2006, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 4435)*, Mitsu Okada and Ichiro Satoh (Eds.). Springer, 272–300. https://doi.org/10.1007/978-3-540-77505-8_23

[22] Evandro Chagas Ribeiro Da Rosa and Rafael De Santiago. 2021. Ket Quantum Programming. *J. Emerg. Technol. Comput. Syst.* 18, 1, Article 12 (Oct. 2021), 25 pages. https://doi.org/10.1145/3474224

[23] Pierre-Luc Dallaire-Demers and Nathan Killoran. 2018. Quantum generative adversarial networks. *Phys. Rev. A* 98 (Jul 2018), 012324. Issue 1. https://doi.org/10.1103/PhysRevA.98.012324

[24] Sumanth Dathathri, Sicun Gao, and Richard M Murray. 2019. Inverse abstraction of neural networks using symbolic interpolation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3437–3444.

[25] Suguru Endo, Zhenyu Cai, Simon C Benjamin, and Xiao Yuan. 2021. Hybrid quantum-classical algorithms and quantum error mitigation. *Journal of the Physical Society of Japan* 90, 3 (2021), 032001.

[26] Yuan Feng and Sanjiang Li. 2023. Abstract interpretation, Hoare logic, and incorrectness logic for quantum programs. *Information and Computation* 294 (2023), 105077. https://doi.org/10.1016/j.ic.2023.105077

[27] Jérôme Feret. 2004. Static Analysis of Digital Filters. In *Programming Languages and Systems, 13th European Symposium on Programming, ESOP 2004, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2004, Barcelona, Spain, March 29 - April 2, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 2986)*, David A. Schmidt (Ed.). Springer, 33–48. https://doi.org/10.1007/978-3-540-24725-8_4

[28] Claudio Ferrari, Mark Niklas Muller, Nikola Jovanovic, and Martin Vechev. 2022. Complete verification via multi-neuron relaxation guided branch-and-bound. *ICRL* (2022).

[29] Irene Gargantini and Peter Henrici. 1971. Circular arithmetic and the determination of polynomial zeros. *Numer. Math.* 18 (1971), 305–320.

[30] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin T. Vechev. 2018. AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*. IEEE Computer Society, 3–18. https://doi.org/10.1109/SP.2018.00058

[31] Roberto Giacobazzi, Isabella Mastroeni, and Elia Perantoni. 2024. Adversities in Abstract Interpretation - Accommodating Robustness by Abstract Interpretation. *ACM Trans. Program. Lang. Syst.* 46, 2 (2024), 5. https://doi.org/10.1145/3649309

[32] Roberto Giacobazzi, Francesco Ranzato, and Francesca Scozzari. 2000. Making abstract interpretations complete. *J. ACM* 47, 2 (2000), 361–416. https://doi.org/10.1145/333979.333989

[33] Edward Grant, Leonard Wossnig, Mateusz Ostaszewski, and Marcello Benedetti. 2019. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *Quantum* 3 (Dec. 2019), 214. https://doi.org/10.22331/q-2019-12-09-214

[34] Vojtech Havlicek, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. 2019. Supervised learning with quantum enhanced feature spaces. *Nature* 567, 7747 (March 2019), 209–212. https://doi.org/10.1038/s41586-019-0980-2 arXiv:1804.11326 [quant-ph].

[35] Kentaro Honda. 2015. Analysis of quantum entanglement in quantum programs using stabilizer formalism. *arXiv preprint arXiv:1511.01572* (2015).

[36] Po-Wei Huang and Patrick Rebentrost. 2024. Post-variational quantum neural networks. arXiv:2307.10560 [quant-ph] https://arxiv.org/abs/2307.10560

[37] A. Yu. Kitaev, A. H. Shen, and M. N. Vyalyi. 2002. *Classical and Quantum Computation.* American Mathematical Society.

[38] Suhas Kotha, Christopher Brix, J Zico Kolter, Krishnamurthy Dvijotham, and Huan Zhang. 2023. Provably bounding neural network preimages. *Advances in Neural Information Processing Systems* 36 (2023), 80270–80290.

[39] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. (2010). http://yann.lecun.com/exdb/mnist

[40] Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, Mykel J Kochenderfer, et al. 2021. Algorithms for verifying deep neural networks. *Foundations and Trends® in Optimization* 4, 3-4 (2021), 244–404.

[41] Sirui Lu, Lu-Ming Duan, and Dong-Ling Deng. 2020. Quantum adversarial machine learning. *Physical Review Research* 2, 3 (2020), 033212.

[42] Luca Marzari, Davide Corsi, Ferdinando Cicalese, and Alessandro Farinelli. 2023. The #DNN-Verification problem: Counting Unsafe Inputs for Deep Neural Networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 217–224.

[43] Luca Marzari, Davide Corsi, Enrico Marchesini, Alessandro Farinelli, and Ferdinando Cicalese. 2024. Enumerating Safe Regions in Deep Neural Networks with Provable Probabilistic Guarantees. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (Eds.). AAAI Press, 21387–21394. https://doi.org/10.1609/AAAI.V38I19.30134

[44] Luca Marzari, Isabella Mastroeni, and Alessandro Farinelli. 2025. Advancing Neural Network Verification through Hierarchical Safety Abstract Interpretation. In *ECAI 2025*. IOS Press.

[45] Isabella Mastroeni. 2024. Abstract domain adequacy. *Int. J. Softw. Tools Technol. Transf.* 26, 6 (2024), 747–765. https://doi.org/10.1007/S10009-024-00774-X

[46] Isabella Mastroeni. 2025. Abstract Local Completeness. In *Verification, Model Checking, and Abstract Interpretation*, Krishna Shankaranarayanan, Sriram Sankaranarayanan, and Ashutosh Trivedi (Eds.). Springer, 3–25.

[47] Andrea Matic, Maureen Monnet, Jeanette Miriam Lorenz, Balthasar Schachtner, and Thomas Messerer. 2022. Quantum-classical convolutional neural networks in radiological image classification. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*. 56–66. https://doi.org/10.1109/QCE53715.2022.00024

[48] Denny Mattern, Darya Martyniuk, Henri Willems, Fabian Bergmann, and Adrian Paschke. 2021. Variational quanvolutional neural networks with enhanced image encoding. *arXiv preprint arXiv:2106.07327* (2021).

[49] Marvin L. Minsky. 1967. *Computation: finite and infinite machines*. Englewood Cliffs, N.J. : Prentice-Hall.

[50] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. 2018. Quantum circuit learning. *Phys. Rev. A* 98 (Sep 2018), 032309. Issue 3. https://doi.org/10.1103/PhysRevA.98.032309

[51] Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. 2009. *Introduction to Interval Analysis*. Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898717716

[52] Mikko Mottonen, Juha J Vartiainen, Ville Bergholm, and Martti M Salomaa. 2004. Transformation of quantum states using uniformly controlled rotations. *arXiv preprint quant-ph/0407010* (2004).

[53] Nidhi Munikote. 2024. Comparing Quantum Encoding Techniques. arXiv:2410.09121 [quant-ph] https://arxiv.org/abs/2410.09121

[54] Michael A. Nielsen and Isaac L. Chuang. 2010. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press.

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[56] Simon Perdrix. 2008. Quantum Entanglement Analysis Based on Abstract Interpretation. In *Static Analysis, 15th International Symposium, SAS 2008, Valencia, Spain, July 16-18, 2008. Proceedings (Lecture Notes in Computer Science, Vol. 5079)*, María Alpuente and Germán Vidal (Eds.). Springer, 270–282. https://doi.org/10.1007/978-3-540-69166-2_18

[57] Luca Pulina and Armando Tacchella. 2010. An Abstraction-Refinement Approach to Verification of Artificial Neural Networks. In *Computer Aided Verification (Lecture Notes in Computer Science, Vol. 6174)*, Tayssir Touili, Byron Cook, and Paul Jackson (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 243–257. https://doi.org/10.1007/978-3-642-14295-6_24

[58] M. Rath and H. Date. 2024. Quantum data encoding: a comparative analysis of classical-to-quantum mapping techniques and their impact on machine learning accuracy. *EPJ Quantum Technology* 11 (2024), 72. https://doi.org/10.1140/epjqt/s40507-024-00285-3

[59] P Richard. 1965. Feynman AR Hibbs,". *Quantum Mechanics and Path Integrals," ISBN-10* 70206503 (1965).

[60] Maria Schuld, Alex Bocharov, Krysta M. Svore, and Nathan Wiebe. 2020. Circuit-centric quantum classifiers. *Phys. Rev. A* 101 (Mar 2020), 032308. Issue 3. https://doi.org/10.1103/PhysRevA.101.032308

[61] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. 2015. An introduction to quantum machine learning. *Contemporary Physics* 56, 2 (2015), 172–185.

[62] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. 2021. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A* 103 (Mar 2021), 032430. Issue 3. https://doi.org/10.1103/PhysRevA.103.032430

[63] Jinjing Shi, Zimeng Xiao, Heyuan Shi, Yu Jiang, and Xuelong Li. 2025. QuanTest: Entanglement-Guided Testing of Quantum Neural Network Systems. *ACM Trans. Softw. Eng. Methodol.* 34, 2, Article 48 (Jan. 2025), 32 pages. https://doi.org/10.1145/3688840

[64] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. 2019. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages* 3, POPL (2019), 1–30.

[65] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[66] John van de Wetering. 2020. ZX-calculus for the working quantum computer scientist. arXiv:2012.13966 [quant-ph] https://arxiv.org/abs/2012.13966

[67] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. 2018. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium (USENIX Security 18)*. 1599–1614.

[68] Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. 2021. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in Neural Information Processing Systems* 34 (2021), 29909–29921.

[69] Tianhao Wei, Hanjiang Hu, Luca Marzari, Kai S Yun, Peizhi Niu, Xusheng Luo, and Changliu Liu. 2025. Modelverification. jl: a comprehensive toolbox for formally verifying deep neural networks. In *Proceedings of the 37th International Conference on Computer Aided Verification*.

[70] Maximilian Wendlinger, Kilian Tscharke, and Pascal Debus. 2024. A comparative analysis of adversarial robustness for quantum and classical machine learning models. In *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, Vol. 1. IEEE, 1447–1457.

[71] Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. 2020. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. *arXiv preprint arXiv:2011.13824* (2020).

[72] Nengkun Yu and Jens Palsberg. 2021. Quantum abstract interpretation. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation* (Virtual, Canada) *(PLDI 2021)*. Association for Computing Machinery, New York, NY, USA, 542–558. https://doi.org/10.1145/3453483.3454061

[73] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. 2018. Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems* 31 (2018).

[74] Xiyue Zhang, Benjie Wang, and Marta Kwiatkowska. 2024. Provable preimage under-approximation for neural networks. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 3–23.