Quantum and Hybrid Machine-Learning Models for Materials-Science Tasks

Leyang Wang[a], Yilun Gong[b,c], Zongrui Pei[a,*]

[a]*New York University, New York, USA*
[b]*Max-Planck-Institut für Nachhaltige Materialien GmbH, Düsseldorf, Germany*
[c]*Department of Materials, University of Oxford, Oxford, UK*

Abstract

Quantum computing has become increasingly practical in solving real-world problems due to advances in hardware and algorithms. In this paper, we aim to design, apply, and evaluate quantum machine learning and hybrid quantum-classical models in a few practical materials science tasks, i.e., predicting stacking fault energies and solutes that can ductilize magnesium. To this end, we adopt two different representative quantum algorithms, i.e., quantum support vector machines (QSVM) and quantum neural networks (QNN), and adjust them to our application scenarios. We systematically test the performance with respect to the hyperparameters of selected ansatzes. Eventually, we construct quantum models with optimized parameters for regression and classification that predict targeted solutes based on the elemental volumes, electronegativities, and bulk moduli of chemical elements. We identify a few combinations of hyperparameters that yield validation scores of approximately 90% for QSVM and hybrid QNN in both tasks.

Keywords: machine learning, quantum computer, quantum computing

1. Introduction

Quantum information science is one of the most active research domains at the intersection of physics, computer science, and applied mathematics. A few critical research directions in this domain include (i) identifying new realizations of qubits (topological qubits, superconducting qubits, neutral atoms, trapped ions, magnetic properties of defects like nitrogen vacancies in diamond), (ii) development of error-correction algorithms and methods that enhance the fidelity of qubits, (iii) the accurate control of qubit states (realization of various quantum gates and measurements), (iv) formulation of the fundamental theorems (i.e., no-go theorems, quasi-probability), etc.

The mechanism of classical computers is built upon Boolean logic, and its fundamental building block is known as a flip-flop [1]. The flip-flop has two stable states, which can be read as high or low. Such a form of retaining information has been integrated into almost every electronic device. Although the functionality and architecture have undergone significant changes since its invention, the basic computing units still store binary information. In the 20th century, many scientists had already developed the concept of quantum computing. One well-known source introduces the idea of simulating physics with a computer that exactly captures the physical property of the real world, as opposed to the classical approximations on classical computers. Richard Feynman explained that simulating quantum physics with classical computers was challenging due to the exponential growth in computing resources required [2, 3]. A computer that is capable of storing probabilistic states will overcome the resource constraint. These ideas that Feynman proposes closely align with the current quantum computers.

One of the major hardware components of quantum computers is quantum bits or qubits, which can be realized by various physical entities [4], such as photons [5], trapped ions [6], superconductors [7], bosons [8], etc. Each type of qubit has its advantages and disadvantages in terms of scalability, operating temperature, decoherence time, and other factors. There has been a surge of updates on the development of quantum hardware from research institutions as well as technology companies that have traditionally built their products using classical computing. This marks a growing interest and greater confidence that quantum computing will serve essential roles in the future. For example, a Google team constructed their quantum computer, Willow, with 105 superconducting qubits [9]. King *et al.* implemented coherent quantum annealing in a programmable 2,000 qubit Ising chain [10] and performed quantum critical dynamics in a 5,000-qubit programmable spin glass [11]. Recently, the team demonstrated supremacy in a practical physical problem, specifically the dynamics of two-, three-, and infinite-dimensional spin glasses, unlike previous studies that focused on idealized systems [12]. In parallel, a Chinese team has transmitted quantum-encrypted images of a record 12,900 kilometers [13].

Machine learning and quantum computers are two technologies that will likely fundamentally change our future. Combining the two, quantum machine learning (QML) has emerged as a novel method with applications in various research domains [14, 15, 16, 17, 18] (see the literature review of QML from 2017 to 2023 [14]). For example, QML has the potential to revolutionize language models [19, 20], and there are already a few initial works to explore this potential [21, 17].

*Corresponding author.
Email addresses: peizongrui@gmail.com (Zongrui Pei), zp2137@nyu.edu (Zongrui Pei)

QML algorithms can be grouped into hybrid quantum/classical algorithms and purely quantum algorithms. Hybrid algorithms, such as quantum support vector machines (QSVM), utilize a quantum circuit as a kernel to replace classical kernels. This group of algorithms leverages both classical and quantum hardware, thereby benefiting from the strengths of both. In contrast, purely quantum algorithms, such as quantum neural networks (QNN), rely solely on quantum hardware. Recent theoretical works show that QNNs exhibit similar behavior to their classical counterparts, which may explain their performance; specifically, they form Gaussian processes [22]. Both hybrid and quantum models can be optimized using classical methods, which are preferred since they are more stable and consistent toward the optimal solutions. Quantum optimization algorithms, such as quantum gradient descent, exhibit probabilistic behavior. Several studies have already implemented QML models to solve real-world problems, such as QSVM models for alloy design [15] and QNN models for supply chain logistics [23].

Magnesium and its alloys are among the lightest structural materials. Once widely utilized in the automotive and aerospace industries, they have the potential to improve energy and environmental sustainability. However, their wide application suffer from easy oxidation and limited ductility. The limited ductility is partly due to insufficient active deformation modes, which is closely related to the stacking fault energies [24, 25, 26, 27]. We will explore how to add solutes to magnesium to improve their ductility, a scientific problem that has received wide attention.

Numerous studies have chosen machine learning methods to build models to predict various materials properties for materials design [28, 29, 20, 19]. Nonetheless, few studies have adopted quantum computing and quantum kernels for such practical tasks. We know very little about the performance of quantum machine learning models in these applications. In this study, we choose the QSVM and QNN primitives to build quantum circuits for QML to solve a specific material-science problem, i.e., predicting the stacking fault energies in hexagonal close-packed magnesium alloys. More specifically, we focus on the intrinsic I_1 stacking faults that form as a result of the dissociation of non-basal dislocations. The performance of these algorithms and models relies on highly optimized quantum circuits. We will focus on a parametric comparison and optimization of these models. We will optimize the quantum models in both classification and regression tasks.

2. Computational Details

2.1. Methods

A general workflow of this study is shown in Figure 1. First, the classical data are preprocessed and tabulated in a structured form. The data consist of three features (bulk modulus B, elemental volume V, electronegativity ν) and one label (stacking fault energies). In classification tasks, the label is converted into 0 and 1. When the value is larger than that of pure magnesium (19 mJ/m^2), then the label is 0; otherwise, the label is 1. The second step is to select one quantum algorithm, such as a

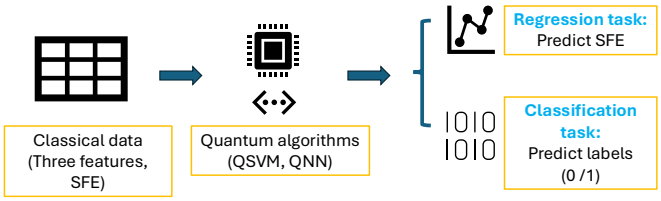


Figure 1: The workflow of this study. It consists of data preprocessing (e.g., normalization, preparation of training and validation data), data encoding into the quantum space, model training, and validation.

QSVM or a QNN. The classical data is first encoded into quantum space and then, depending on a specific algorithm, further processed to predict the labels. Similar processes are implemented for both the regression and classification tasks.

To execute QML, we require coding libraries that orient toward both classical and quantum machine learning algorithms. Commonly used quantum computing software includes Qiskit [30], Cirq [31], Pennylane [32], TensorFlow Quantum [33], etc. Our method contains both classical and quantum parts. Our quantum SVM and QNN models are from Qiskit [30]; the classical data relies on the SciKit-Learn [34] package.

2.1.1. Quantum Support Vector Machine

The classical kernel of the SVM algorithm is replaced with a quantum kernel, which is computed by a quantum circuit. We design a quantum circuit with multiple qubits and various quantum gates, such as the n-qubit quantum circuit in Figure 2 (upper panel, $n = 3$). All n qubits are set to an initial state $|0\rangle^{\otimes n}$, and then various gates are added to manipulate the states. Standard quantum gates include rotation gates, Pauli gates, and, most importantly, the gates involving multiple qubits to introduce quantum entanglement. More specifically, we apply the unitary operation $U_{\Phi(\boldsymbol{x})}$ to the initial state for the data sample $\boldsymbol{x}$, where entanglement is included. Then we have

$$|\Phi(\boldsymbol{x})\rangle = U_{\Phi(\boldsymbol{x})}|0\rangle^{\otimes n}, \tag{1}$$

where $U_{\Phi(\boldsymbol{x})} = \prod_l u_{\Phi(\boldsymbol{x})} H^{\otimes n}$. Here, H is the Hadamard gate and

$$u_{\Phi(\boldsymbol{x})} = \exp\left(i \sum_{S\in[n]} \phi_S(\boldsymbol{x}) \prod_{i\in S} P_i\right). \tag{2}$$

The quantum gates $P_i \in \{I, X, Y, Z\}$ are the Pauli gates [35], and the index S denotes the connectivity between qubits or data points. More details about the quantum gates can be found in Ref. [15]. Since l determines the number of repetitions of the quantum gates $u_{\Phi(\boldsymbol{x})}H^{\otimes n}$, it is also referred to as the depth of the quantum circuit. The functions $\phi_{\{i\}}(\boldsymbol{x}) = x_i$ and $\phi_{\{1,2\}}(\boldsymbol{x}) = (\pi - x_1)(\pi - x_2)$ for $|S| = 1$ and $|S| = 2$, respectively. Here, the symbol $|S|$ represents the number of qubits involved. We need $|S| = 2$ to entangle the qubits. Assisted by quantum circuits, we map one data sample $\boldsymbol{x}$ in the training data to a quantum state $\Phi(\boldsymbol{x})$, i.e., $\boldsymbol{x} \in X \rightarrow |\Phi(\boldsymbol{x})\rangle\langle\Phi(\boldsymbol{x})|$. This procedure encodes the training data X into the quantum space. After

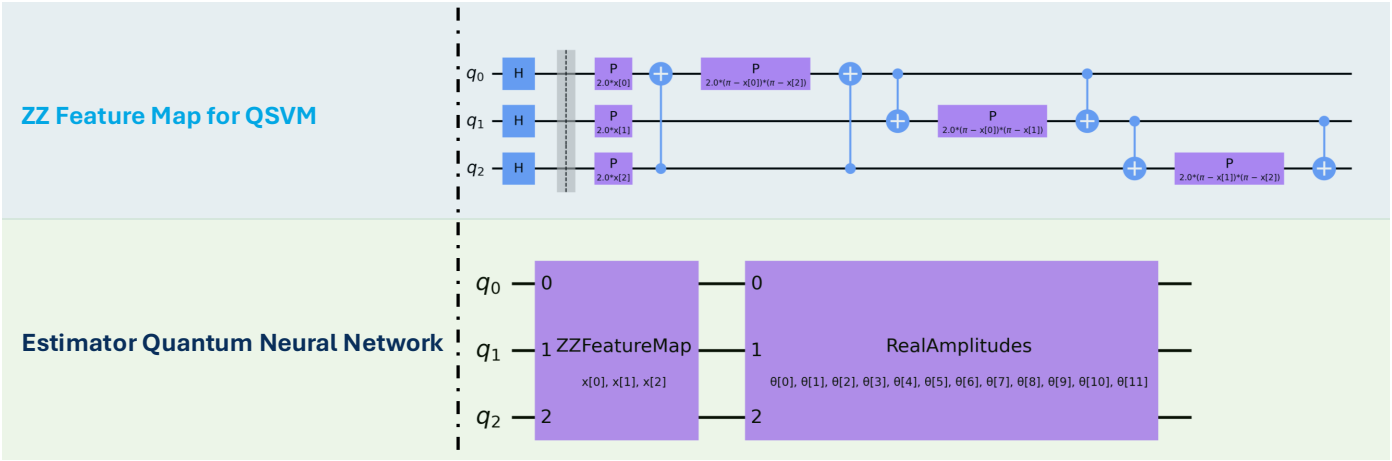


Figure 2: The two algorithms and their representative quantum circuits. Since the data in this study has three features, we only need three qubits. The first quantum circuit is the kernel in a quantum vector machine (QSVM). The second quantum circuit uses the quantum neural network (QNN) model.

the data encoding, quantum circuits can provide a quantum kernel $K(\boldsymbol{x}, \boldsymbol{x}') = |\langle\Phi(\boldsymbol{x})|\Phi(\boldsymbol{x}')\rangle|^2$ needed for our quantum SVM models.

The quantum SVM algorithm adopted here is similar to the classical SVM algorithm except that quantum kernels or quantum circuits replace classical kernels in the classical SVM. So, we provide a brief introduction to the classical SVM and then describe how the quantum kernels and circuits are used. The target of an SVM model is to find a hyperplane like $y = wX$ that meets the criterion

$$\text{sign}(wX - y) \in [-1, 1] \tag{3}$$

for given dataset $\{X, y\}$, where w is a weight matrix. The optimal hyperplane maximizes the separation of the data with different labels. The hyperplane perpendicular to the shortest distance between the two groups of subsets is an ideal choice. Mathematically, finding this hyperplane is equivalent to minimizing the reciprocal inverse magnitude of the weight vector w, i.e.,

$$\min\left\{\frac{1}{\|w\|}\right\}. \tag{4}$$

Before we perform the minimization, we need to choose a kernel $K(\boldsymbol{x}, \boldsymbol{x}')$. The kernel maps the data from the original space to a new space. It measures the correlation between data points. In prediction, the kernel acts as a weight to determine the value y for an unknown X. The closer the two points are, the higher the weight. The choice of a kernel is critical for the performance of an ML model. A well-chosen kernel can significantly ease the separation of different classes in this high-dimensional space. A kernel can be linear or non-linear, depending on the data distribution. For data with a complicated pattern, classical functions that can be expanded as polynomials may not be the best choice; instead, the quantum kernel, which incorporates quantum correlation and entanglement between qubits, offers a unique advantage. The guiding principle is that the more challenging it is to express the chosen kernel classically, the greater the benefit achieved. For example, a polynomial can hardly describe the quantum kernel with a high level of entanglement. Figure 2 (upper panel) shows an example of the quantum circuit used in our quantum SVM algorithm. We employ several different quantum circuits and tune their hyperparameters to find those that offer optimal performance. We have used the method in designing CCAs, where more qubits are involved [15].

2.1.2. Quantum Neural Network and Hybrid Model

Qiskit currently offers a number of QNN implementations for different purposes. In this study, we construct algorithms using both the EstimatorQNN and the SamplerQNN object and optimize the hyperparameters involved. An example of EstimatorQNN is shown in Figure 2 (lower panel). EstimatorQNN used a feature map like the entangled ZZFeatureMap [30] to encode the classical data into the quantum space. This step is the same as QSVM. Differently, QNN has a higher level of involvement in quantum circuits. Unlike the hybrid QSVM, which outsources the rest of the algorithm to a classical algorithm, QNN continues to utilize another quantum ansatz (e.g., RealAmplitudes) to process the data and complete classification or regression tasks.

Qiskit supports a hybrid version of QNN where the QNN becomes part of an ANN defined by PyTorch. In a hybrid QNN, the entire circuit is used as a layer of the classical neural network. More specifically, it wraps the QNN into a PyTorch module using TorchConnector. This can increase the nonlinearity of the NN and improve its accuracy. More importantly, this connector module allows users to use mature practices from classical machine learning. We will provide a setup for the hybrid QNN method in this study, but we have omitted attempts to customize the forward function or loss functions. For more details of the hybrid model, please refer to `https://qiskit-community.github.io/qiskit-machine-learning/tutorials/05_torch_connector.html`.

2.2. Dataset

The dataset used in this study is three elemental properties and the stacking fault energies (SFEs) when these elements ap-

Table 1: QSVC Hyperparameter Options

Hyperparameter	Type	Range
C	Discrete	[0.1, 1, 10, 100]
entanglement	Categorical	[circular, full, linear]
reps	Discrete	[1, 2, 3, 4, 5]

Table 2: QSVR Hyperparameter Options

Hyperparameter	Type	Range
C	Discrete	[0.1, 1, 10, 100]
entanglement	Categorical	[circular, full, linear]
reps	Discrete	[1, 2, 3, 4, 5]

pear in magnesium with a concentration of 6.25 at.% [25]. The three elemental properties, including the bulk modulus (B in gigapascals), atomic volume (V in Å^3), and electronegativity (ν, unit-less), are taken as features, meaning a maximum of three qubits is needed. The SFEs are used as the target values in regression models. In classification models, we label all chemical elements with an SFE lower than 19 mJ/m^2 as 0 and the rest as 1. The bulk modulus of Tc is missing in Ref. [36], which is about 281 GPa from `https://www.matweb.com/search/datasheet_print.aspx?matguid=2629eceea53a4c0dbb9279fac6e6007d`

3. Results and Discussion

3.1. Quantum SVM for Classification

We perform hyperparameter tuning and limit the range for some important hyperparameter in Table 1. For this experiment, we adopt the common technique of k-fold cross-validations, here the number of folders is the size of the dataset divided by a test size of three. Within each fold, the shuffling train-test split uses 80% of data for training and 20% for validation. The final result is averaged over three tests. The hyperparameter tested are:

- C: balances the accuracy of the model and the width of the margin in SVM. For example, a smaller value of C means a larger margin and can tolerate more misclassifications. We examined a range of C values corresponding to powers of ten, ranging from 0.01 to 100.
- reps: also called the depth of quantum circuit, usually represented by l. It is specific to quantum machine learning and means the number of repetitions for the quantum circuit. The value must be an integer larger than or equal to 1.
- entanglement: specific to the quantum machine library, and specifies which entanglement options to use.

When C is 1, 10, or 100, the accuracy is higher on average (Figure 3). The value of gamma and the entanglement option do not show an obvious correlation with the test accuracy. The optimal performance of 0.92 is achieved for $C = 1$, reps=3, and full entanglement, which reduces marginally to 0.91 when we use the linear entanglement, keeping all other parameters identical. This indicates that a higher level of entanglement yields improved results, albeit weakly. A comparable performance is observed for $C = 100$, reps=1, and circular entanglement.

3.2. Quantum SVM for Regression

Similar to the classification task, we perform parameter tuning in C, reps, and entanglement to find out the most optimal parameter settings for the given task (Table 2). For this experiment, we compare the average result after 20 iterations of shuffled train-test split groups.

From Figure 4, we observe that performance in most cases is worse when $C = 0.1$. In addition, the model performance drops at higher reps, accounting for circular and full entanglement, while the linear entanglement suggests a different trend. The optimal performance is found for reps=1 for the three types of entanglement, regardless of the choices for other parameters. More specifically, when we use circular or full entanglement, the optimal performance is 0.88 when $C = 1, 10, 100$. The accuracy reduces to 0.8 when $C = 0.1$. The accuracy decreases substantially to 0.81 when we switch the entanglement from full/circular to linear types. This apparent change clearly shows the importance of using high-level entanglement.

3.3. Hybrid Quantum Neural Network for Classification

We first use QNN on a classification task and consider key hyperparameters, including the number of repeats for the feature map, ansatz, and entanglement methods. The classification rubric is the same as QSVC, where each element is assigned to be 0 or 1 depending on whether its SFE is greater than 19 mJ/m^2. A detailed implementation of the algorithm is provided in the Code Availability section. Figure 5 shows the QNN results for classification with various repeats or depths l of core quantum circuits. The three horizontal bar graphs represent the correctness of the three repeats. The accuracy is 0.810 for $l = 1$, 0.905 for $l = 2$, and 0.857 for $l = 3$. These results indicate that the optimal repeat of the core circuits is $l = 2$. Specifically, Pr and Sc are improperly predicted across all three repeats. When $l = 1$, the labels of Ti and Zr are also improperly predicted. When $l = 3$, Pr, Sc, and Ti are mislabeled. Therefore, the best performance or accuracy is obtained for $l = 2$.

3.4. Hybrid Quantum Neural Network for Regression

We adopted a hybrid model in which the QNN is wrapped in a PyTorch configuration for classical ML, enabling better control over neural network layers by adding an activation layer or customizing the loss function. We consider three different depths l or a repeat of the quantum circuit: 1, 2, and 3. The linear trend is evident in Figure 6. Overall, it follows the y = x line with four chemical elements as outliers. Nd and Pr correspond to the smallest SFEs, making it difficult to predict; Os and Co are two non-rare-earth elements that have SFEs larger

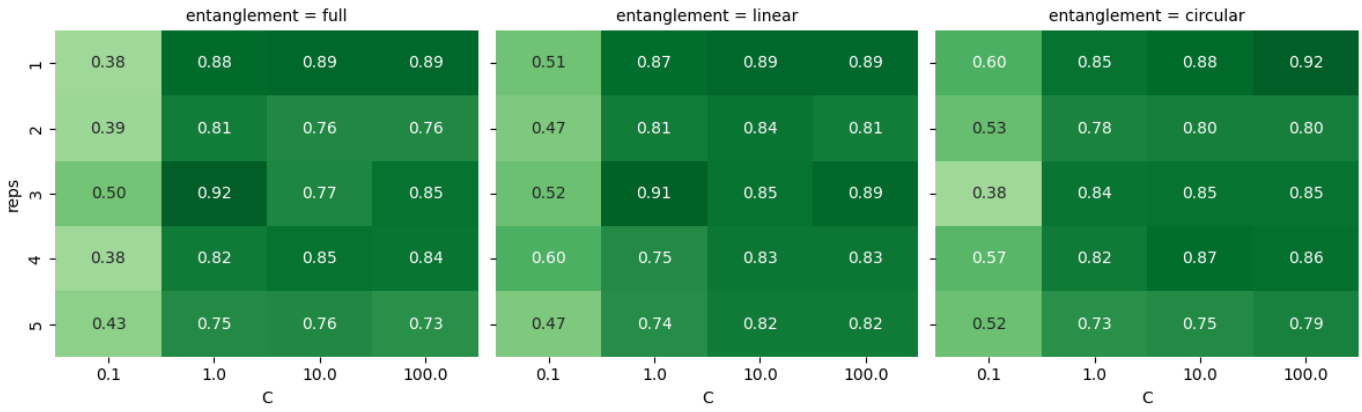


Figure 3: Validation scores of QSVC. Here, the validation scores with various C, $reps$, and *entanglement* options are considered.

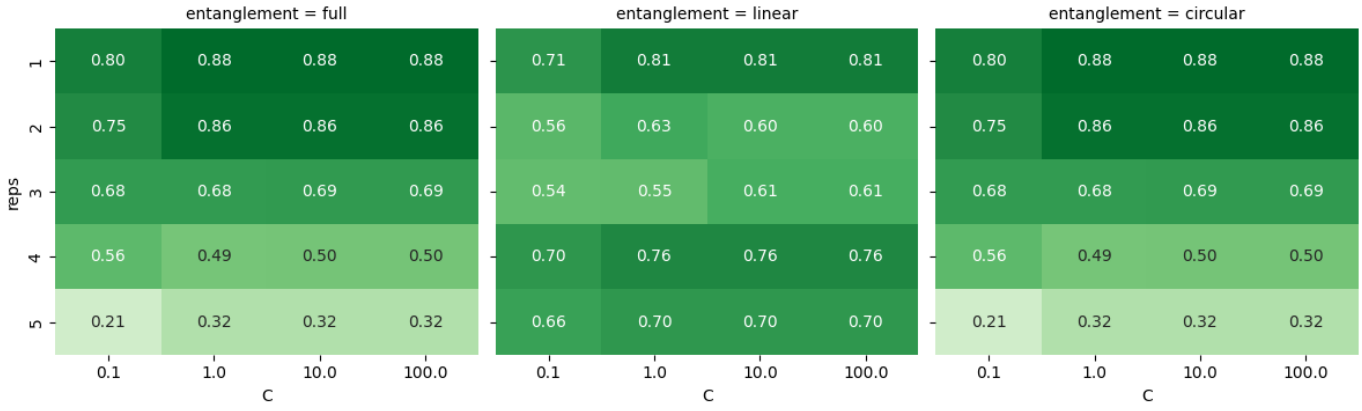


Figure 4: Validation scores of QSVR. Here, the validation scores with various C, $reps$, and *entanglement* options are considered.

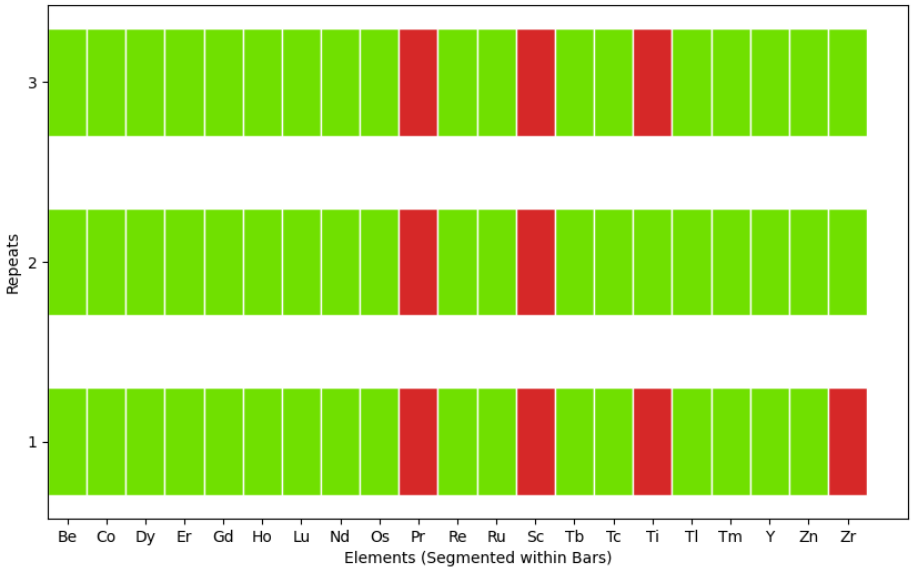


Figure 5: The hybrid QNN model for classification. The three horizontal bar graphs represent the correctness of the three repeats. Pr and Sc are wrongly predicted across all three repeats. The accuracy is 0.810 for $l = 1$, 0.905 for $l = 2$, and 0.857 for $l = 3$.

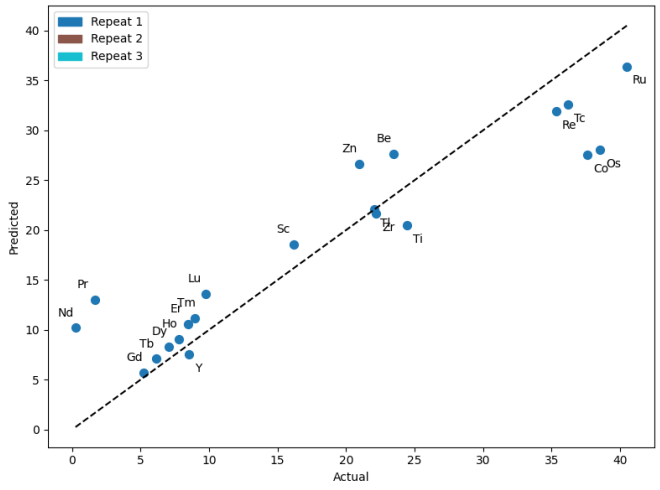


Figure 6: The hybrid QNN model for regression. Here, the horizontal and vertical axes represent the actual and QNN-predicted SFEs for different repeats or depths l of the quantum circuit. Left side outliers: Nd, Pr, whose SFEs are the lowest; and right side outliers are Os and Co. Pearson's R^2 is 0.875 for repeat 1, 0.865 for repeat 2, and 0.864 for repeat 3.

than 35 mJ/m^2 and are predicted to be around 25 mJ/m^2. Since this number is larger than the SFE of pure Mg (~19 mJ/m^2), this prediction groups them into the solutes that cannot ductilize Mg, which is qualitatively correct.

3.5. Future Work

We have implemented two types of QML algorithms in classification and regression tasks. Our goal for the future is to introduce more profuse quantum anzates or circuits in the context of machine learning, and systematically test their performance. These methods may or may not be applicable in QML, and their performance in solving real materials science tasks is unclear, making them valuable to the materials science community. One targeted algorithm is the variational quantum regressor/classifier, a representative method in the variational approach. The target is to minimize the error function $L(\boldsymbol{\theta})$ by finding the optimal parameters $\boldsymbol{\theta}$, where L is defined by

$$\min L(\boldsymbol{\theta}),\ L(\boldsymbol{\theta}) = \langle\psi|U^{\dagger}(\boldsymbol{\theta})HU(\boldsymbol{\theta})|\psi\rangle, \tag{5}$$

where the parameters $\boldsymbol{\theta}$ will be determined at the minimal error function for all given wavefunctions. One part of the quantum circuit is used to implement the parameters $\boldsymbol{\theta}$, while a separate part encodes the input data X. To accelerate quantum computing, either in quantum machine learning or quantum chemistry simulations, GPU acceleration will be implemented, where multiple quantum circuits corresponding to different $\boldsymbol{\theta}$ values are computed simultaneously. In addition, we hope to fully leverage the flexibility of hybrid QNN by including customized functions.

The ultimate goal of this series of studies is to identify the common features or patterns that are likely to lead to successful high-performance QML models for practical tasks rather than idealized, simplified tasks based on synthesized data.

4. Conclusions

Quantum machine learning models are usually tested on synthetic or idealized data in previous studies. Using data from a real materials-science problem, we examine the performance of

two quantum algorithms: one quantum algorithm and one hybrid quantum algorithm, specifically the quantum support vector machine and the Estimator quantum neural network. We developed models for both classification and regression tasks using these algorithms. In the quantum support vector machine for the classification task, an optimal accuracy of 0.92 is achieved by three fully entangled qubits in a quantum circuit with a depth of $l = 3$. In the quantum support vector machine for the regression task, an optimal accuracy of 0.88 is achieved by three fully or circularly entangled qubits in a quantum circuit with a depth of $l = 1$. In the hybrid quantum neural network model for the classification task, an optimal accuracy of 0.91 is achieved by a three-qubit quantum circuit with a depth of $l = 1$. In the quantum support vector machine for the regression task, an optimal accuracy of 0.88 is achieved by a three-qubit quantum circuit with a depth of $l = 1$. Our results confirmed that even with the same data and algorithm, the best model is obtained with very different hyperparameters. This result confirms the complicated fact that we need to tailor the hyperparameters to get an optimal model, even for the same data and algorithm in different tasks. Overall, complex entanglement states, either circular or fully entangled, are superior to non-entangled states. The promising accuracies of these four models indicate their applicability in real materials science problems. Our study offers valuable insights into the applications of quantum machine learning in addressing real-world problems.

Code Availability

The code used in this study is available on GitHub: The version of code that uses fivefold cross-validations: `https://github.com/lw3266/QML-SFEPrediction`; the version of code that uses twentyfold cross-validations: `https://github.com/gongyilun/QMML_SFE_Mg/tree/main`. To help reproduce the results, we provide a list of the specific packages and their versions below: first create a python environment with *Python 3.11.4*; then, pip install *qiskit==0.46.3 qiskit-aer==0.12.1 qiskit-algorithms==0.3.1 qiskit-ibm-provider==0.6.1 qiskit-machine-learning==0.7.2 qiskit-terra==0.46.3 qiskit 0.46.3 qiskit-aer 0.12.1 qiskit-algorithms 0.3.1 qiskit-ibm-provider 0.6.1 qiskit-ibmq-provider 0.20.2 qiskit-machine-learning 0.7.2 qiskit-terra 0.46.3*.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

[1] N. Burla, Introduction to Flip Flop, OER Commons, 2016. `https://oercommons.org/authoring/15863-introduction-to-flip-flop`.

[2] R. P. Feynman, Simulating Physics with Computers, in: Feynman and Computation, CRC Press, 2018, pp. 133–153.

[3] R. P. Feynman, Quantum Mechanical Computers, Foundations of Physics 16 (6) (1986) 507–532.

[4] T. Confalone, F. Lo Sardo, Y. Lee, et al., Cuprate Twistronics for Quantum Hardware, Advanced Quantum Technologies 8 (11) (2025) 2500203.

[5] H. Aghaee Rad, T. Ainsworth, R. Alexander, et al., Scaling and Networking a Modular Photonic Quantum Computer, Nature 638 (8052) (2025) 1–8.

[6] M. Liu, R. Shaydulin, P. Niroula, et al., Certified Randomness Using a Trapped-Ion Quantum Processor, Nature 640 (8058) (2025) 1–6.

[7] C. D. Wilen, S. Abdullah, N. Kurinsky, et al., Correlated Charge Noise and Relaxation Errors in Superconducting Qubits, Nature 594 (7863) (2021) 369–373.

[8] H. Putterman, K. Noh, C. T. Hann, et al., Hardware-Efficient Quantum Error Correction via Concatenated Bosonic Qubits, Nature 638 (8052) (2025) 927–934.

[9] R. Acharya, D. A. Abanin, L. Aghababaie-Beni, et al., Quantum Error Correction Below the Surface Code Threshold, Nature 638 (8052) (2024) 920–926.

[10] A. D. King, S. Suzuki, J. Raymond, et al., Coherent Quantum Annealing in a Programmable 2,000-Qubit Ising Chain, Nature Physics 18 (11) (2022) 1324–1328.

[11] A. D. King, J. Raymond, T. Lanting, et al., Quantum Critical Dynamics in a 5,000-Qubit Programmable Spin Glass, Nature 617 (7959) (2023) 61–66.

[12] A. D. King, A. Nocera, M. M. Rams, et al., Beyond-Classical Computation in Quantum Simulation, Science 388 (6743) (2025) eado6285.

[13] Y. Li, W.-Q. Cai, J.-G. Ren, et al., Microsatellite-Based Real-Time Quantum Key Distribution, Nature 640 (8057) (2025) 471–478.

[14] D. Peral-García, J. Cruz-Benito, F. J. García-Peñalvo, Systematic Literature Review: Quantum Machine Learning and Its Applications, Computer Science Review 51 (2024) 100619.

[15] Z. Pei, Y. Gong, X. Liu, J. Yin, Designing Complex Concentrated Alloys with Quantum Machine Learning and Language Modeling, Matter 7 (10) (2024) 3433–3446.

[16] Z. Pei, Computer-Aided Drug Discovery: From Traditional Simulation Methods to Language Models and Quantum Computing, Cell Reports Physical Science 5 (12) (2024) 102334.

[17] J. Del Castillo, D. Zhao, Z. Pei, Comparative Study of the Ansätze in Quantum Language Models, arXiv preprint arXiv:2502.20744 (2025).

[18] X. Liu, J. Zhang, Z. Pei, Machine Learning for High-Entropy Alloys: Progress, Challenges, and Opportunities, Progress in Materials Science 131 (2023) 101018.

[19] Z. Pei, J. Yin, J. Zhang, Language Models for Materials Discovery and Sustainability: Progress, Challenges, and Opportunities, Progress in Materials Science 154 (2025) 101495.

[20] Z. Pei, J. Yin, J. Neugebauer, A. Jain, Towards the Holistic Design of Alloys with Large Language Models, Nature Reviews Materials 9 (12) (2024) 840–841.

[21] G. Buonaiuto, R. Guarasci, A. Minutolo, G. De Pietro, M. Esposito, Quantum Transfer Learning for Acceptability Judgements, Quantum Machine Intelligence 6 (1) (2024) 13.

[22] D. García-Martín, M. Larocca, M. Cerezo, Quantum Neural Networks Form Gaussian Processes, Nature Physics 21 (2025) 1153–1159.

[23] R. Correll, S. J. Weinberg, F. Sanches, T. Ide, T. Suzuki, Quantum Neural Networks for a Supply Chain Logistics Application, Advanced Quantum Technologies 6 (7) (2023) 2200183.

[24] Z. Pei, L.-F. Zhu, M. Friák, et al., Ab Initio and Atomistic Study of Generalized Stacking Fault Energies in Mg and Mg–Y Alloys, New Journal of Physics 15 (4) (2013) 043020.

[25] S. Sandlöbes, Z. Pei, M. Friák, et al., Ductility Improvement of Mg Alloys by Solid Solution: Ab Initio Modeling, Synthesis, and Mechanical Properties, Acta Materialia 70 (2014) 92–104.

[26] Z. Pei, M. Friák, S. Sandlöbes, et al., Rapid Theory-Guided Prototyping of Ductile Mg Alloys: From Binary to Multi-Component Materials, New Journal of Physics 17 (9) (2015) 093009.

[27] Z. Pei, J. Yin, Machine Learning as a Contributor to Physics: Understanding Mg Alloys, Materials & Design 172 (2019) 107759.

[28] D. Zhao, J. Qiao, Y. Zhang, P. K. Liaw, Cooperative Game During Phase Transformations in Complex Alloy Systems, Scripta Materialia 257 (2025) 116440.

[29] Z. Pei, K. A. Rozman, Ö. N. Doğan, et al., Machine-Learning Microstructure for Inverse Material Design, Advanced Science 8 (23) (2021) 2101207.

[30] A. Javadi-Abhari, M. Treinish, K. Krsulich, et al., Quantum Computing with Qiskit, arXiv preprint arXiv:2405.08810, 2024.

[31] Cirq Developers, Cirq (v1.6.1), Zenodo, 2025. `https://doi.org/10.5281/zenodo.16867504`.

[32] V. Bergholm, J. Izaac, M. Schuld, et al., Pennylane: Automatic differentiation of hybrid quantum-classical computations, arXiv preprint arXiv:1811.04968, 2018.

[33] M. Broughton, G. Verdon, T. McCourt, et al., Tensorflow Quantum: A Software Framework for Quantum Machine Learning, arXiv preprint arXiv:2003.02989 (2020).

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[35] V. Havlíček, A. D. Córcoles, K. Temme, et al., Supervised Learning with Quantum-Enhanced Feature Spaces, Nature 567 (7747) (2019) 209–212.

[36] T. Gray, N. Mann, M. Whitby, Periodic table, `http://periodictable.com`, 2017. Accessed: 2023-12-18.