GNN-Based Performance Prediction of Quantum Optimization of Maximum Independent Set

Special Session Paper

Atefeh Sohrabizadeh
University of California, Los Angeles
Los Angeles, CA 90095, USA

Wan-Hsuan Lin
University of California, Los Angeles
Los Angeles, CA 90095, USA

Daniel Bochen Tan
University of California, Los Angeles
Los Angeles, CA 90095, USA

Madelyn Cain
Department of Physics
Harvard University
Cambridge, MA 02138, USA

Sheng-Tao Wang
QuEra Computing Inc.
Boston, MA 02135, USA

Mikhail D. Lukin
Department of Physics
Harvard University
Cambridge, MA 02138, USA

Jason Cong
University of California, Los Angeles
Los Angeles, CA 90095, USA

ABSTRACT

Maximum Independent Set (MIS) is an NP-hard optimization problem with wide-ranging applications in science and technology. Recently, a super-linear speedup over classical simulated annealing in solving MIS was experimentally observed using a Rydberg atom array (RAA) quantum computer. The extent of the observed speedup depended on the graph instance and the circuit depth of the quantum algorithm. Due to the limited availability of RAA, it is beneficial to be able to efficiently predict the quantum optimization performance on a given graph and circuit depth prior to running it. In this work, we present a graph neural network (GNN)-based performance predictor of the RAA-based MIS optimizer. Our experimental results achieve accuracy with an average root mean squared error (RMSE) of 0.03 out of the range [0, 1]. We open source the experimental data collected for this study at https://github.com/UCLA-VAST/RAAMIS.

ACM Reference Format:
Atefeh Sohrabizadeh, Wan-Hsuan Lin, Daniel Bochen Tan, Madelyn Cain, Sheng-Tao Wang, Mikhail D. Lukin, and Jason Cong. 2024. GNN-Based Performance Prediction of Quantum Optimization of Maximum Independent Set: Special Session Paper. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '24), October 27–31, 2024, New York, NY, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3676536.3697129

1 INTRODUCTION

A major goal in quantum information science has been to show quantum advantage on a problem with useful practical applications. Combinatorial optimization problems, which seek to minimize a cost function over bit strings, have wide-ranging applications in science and technology. These problems also form the foundation of modern computer science and solving them exactly is NP-hard [18]. Quantum combinatorial optimization algorithms that can be implemented on quantum devices in the near-term typically involve adiabatic [9] or variationally parametrized [8] quantum evolution optimized by closed-loop classical feedback [19, 38].

While these algorithms are guaranteed to solve the problem in the limit of infinite circuit depth, less is known about their finite-depth performance despite nearly two decades of research. Theoretical studies are fundamentally limited to small system sizes [42] or shallow circuits [10] due to the inherent intractability of classically simulating quantum systems. Prior experimental studies also are limited to small system sizes [14, 26] or lack significant quantum coherence beyond shallow circuit depths [12, 15, 32], and as a result, offer only limited insights into the algorithms' performances at large system sizes and high circuit depths, the regime believed to be necessary for quantum advantage [4, 7].

Ebadi *et al.* [6] addresses this problem by experimentally implementing quantum algorithms for solving Maximum Independent Set (MIS), a paradigmatic NP-hard optimization problem [18], using a Rydberg atom array (RAA) quantum computer (a similar RAA is now commercially available [29, 40]). The goal of MIS is to find the maximum independent set of a graph, i.e., the largest subset of nodes where no pair of nodes are connected by an edge. By utilizing a hardware-efficient encoding associated with the Rydberg blockade mechanism, this work reached system sizes of hundreds of qubits at relatively high circuit depths of ~ 32. Ebadi *et al.* experimentally observed a super-linear quantum speedup over classical simulated annealing (SA) in solving MIS on the hardest graph instances for SA. The extent of the quantum speedup, however, depended on both the graph instance and the circuit depth of the quantum algorithm [3].

Motivated by this work, we present a graph neural network (GNN)-based performance predictor, MIS-GNN, for the RAA-based MIS optimizer. We take a data-driven approach by training neural networks to predict quantum performance because of the difficulty in simulating quantum systems exactly. We choose GNN since it can directly take the graph instance as the input. Additionally, GNNs

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
ICCAD '24, October 27–31, 2024, New York, NY, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1077-3/24/10.
https://doi.org/10.1145/3676536.3697129

have been shown to be effective in predicting complex performance metrics in electronic design automation, e.g., [2, 13, 20, 31, 34].

Specifically, we build a bi-directional graph for each instance and define appropriate attributes for both the nodes and edges to help the model better learn the graph features (i.e., embeddings). The nodes take the geographical locations and the circuit depth as the attributes while the edges are divided into eight types to identify the whereabouts of the neighbors. The graph is passed through five layers of TransformerConv [33], a state-of-the-art GNN encoding approach for the node embeddings. The final node embeddings are chosen by exploiting a jumping knowledge network architecture [41] to dynamically adjust the ranges of the neighborhood for each node. Then, we concatenate the aggregated results of the node embeddings, which create a graph-level embedding, with the circuit depth value and pass them through multi-layer perceptron (MLP) networks to predict our final objectives: $P_{|\mathrm{MIS}|}$ (MIS probability) and R (approximation ratio). $P_{|\mathrm{MIS}|}$ is the probability of the quantum algorithm finding an MIS of the graph. R is the expectation value of the size of the independent set found by the quantum algorithm divided by $|\mathrm{MIS}|$, the true size of MIS of the graph.

The organization of the paper is as follows: in Sec. 2, we cover the background on optimizing MIS with RAA, the hardness of predicting quantum MIS performance, and the principles of GNN; in Sec. 3, we detail the encoding and architecture of the GNN model; in Sec. 4, we present evaluations on the accuracy of the GNN predictor; in Sec. 5, we conclude the paper and discuss future directions.

2 BACKGROUND

2.1 Unit-Disk MIS on Rydberg Atom Arrays

Rydberg atom arrays can solve a wide variety of combinatorial optimization problems using hardware-efficient encodings [25]. Motivated by recent experiments [6], we focus on MIS. RAA naturally encodes MIS on certain unit-disk graphs, where the nodes are on a 2D grid, and an edge connects each pair of nodes if and only if this pair is located within a unit radius, as shown in Fig. 1a.

To solve MIS on such a unit-disk graph $G = (V, E)$ using RAA [6, 28], we can associate each node with a single atom and make use of two atomic states: the ground state, $|0\rangle$, and a highly excited state, $|1\rangle$ called a Rydberg state. We can arrange the atoms in a configuration induced by G and initialize them to $|0\rangle$, as illustrated in Fig. 1a. Then, by applying a quantum adiabatic algorithm [9] we can minimize the energy:

$$H_{\mathrm{Ryd}} = -\delta \sum_{u\in V} n_u + \frac{1}{2} \sum_{u,v\in V, u\neq v} V_{uv} n_u n_v, \tag{1}$$

where $n_u = 0$ if atom u is in $|0\rangle$ and $n_u = 1$ if the atom is in $|1\rangle$, $V_{uv} \propto |\vec{x}_u - \vec{x}_v|^{-6}$ is the interaction energy depending on the distance between u and v, and δ is a tunable parameter.

The interaction energy models the Rydberg blockade mechanism [23]: when both u and v are excited and the distance between them is small, $V_{uv} n_u n_v$ increases rapidly, which is energetically unfavorable. The maximal distance when V_{uv} is still significant is the *Rydberg blockade radius*, R_b, which we choose to be the unit length. Because G is a unit-disk graph, minimizing the quadratic term in H_{Ryd}, $\sum V_{uv} n_u n_v$, means that the adjacent atoms (within R_b apart), are not simultaneously excited. Thus, the set of excited atoms, $\mathcal{A}$, is an independent set. Moreover, as the cardinality of $\mathcal{A}$, $\sum n_u$ in H_{Ryd}, is maximized, the algorithm optimizes $\mathcal{A}$ towards a maximum independent set, as exhibited in Fig. 1b.

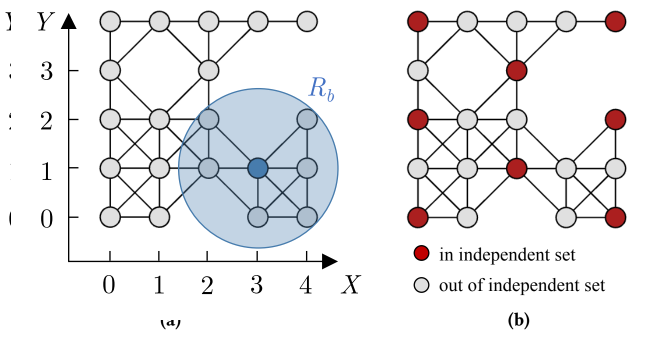

Figure 1: (a) Atoms are initially arranged in a configuration encoding a unit-disk graph, such that two nodes are connected by an edge if and only if they are within the Rydberg blockade radius R_b. All atoms are initialized in $|0\rangle$, corresponding to no nodes in the independent set. (b) After an evolution under the quantum adiabatic algorithm, excited atoms in $|1\rangle$ constitute the MIS solution by the quantum algorithm (red vertices).

We use the setting from Ebadi *et al.* [6] where the grid separation is $R_b/\sqrt{2}$ (see Fig. 1a) so that the nearest and next-nearest neighbors have significant interaction energy. Solving MIS on this particular class of unit-disk graphs is NP-hard and thus general enough to solve any problem in NP [6].

2.2 Hardness of Estimating RAA-Based Quantum Optimizer Performance

Currently, the availability of quantum computers that can run the adiabatic algorithm is still limited. Thus, it is valuable to estimate the performance of the algorithm before actually spending the effort of implementing it on a RAA.

From the principles of the quantum adiabatic algorithm, Ebadi *et al.* [6] reasoned that the probability of such algorithm finding an MIS is $P_{|\mathrm{MIS}|} \simeq 1 - e^{-c' \, d_{\mathrm{QAA}}/g_{\mathrm{QAA}}}$, where d_{QAA} is the *quantum depth* and g_{QAA} is the minimum spectral gap between the ground and first excited multi-atom state during the evolution. The quantum depth is simply the total runtime of the quantum algorithm divided by certain unit time (the time to flip a single atom state). However, in the worst case, exactly computing g_{QAA} take exponential time in the system size [1, 21]. Thus, it is of interest to find other efficient methods to estimate $P_{|\mathrm{MIS}|}$.

Even if $P_{|\mathrm{MIS}|}$ is small, the quantum algorithm may still be able to generate solutions that are close to the optimum. Thus, another relevant performance metric is the approximation ratio, R, which is defined as the observed independent set size generated by the algorithm, divided by the size of the largest independent set, i.e., $|\mathrm{MIS}|$. $R = 1$ if the algorithm outputs the ideal solution. Thus, R describes the ability of an algorithm to find approximate solutions to MIS. Computing R exactly also involves understanding the spectral gaps during the quantum evolution, so this computation is hard for classical computers.

2.3 Graph Neural Networks

Nowadays, graphs are among the core data structures used in data centers. Graph neural networks (GNNs) [39] are developed to extract low-dimensional structured information from graphs, which by nature are unstructured. In other words, GNNs learn to assign representative features (also called embeddings) to the nodes and edges of the graphs that can be helpful in better analyzing them, and, they have been proven to be powerful in many domains ranging from social network analysis [36] to electronic design automation [13, 34]. Because of the efficiency and demonstrated capability of GNNs, we leverage them to *approximately* estimate the performance of RAA-based MIS optimization.

GNNs, in essence, rely on multiple layers of 'message passing' which updates the embeddings of each node by gathering the information from its neighboring nodes/edges. Layer l of a GNN can be formulated as:

$$\boldsymbol{h}_i^l = \sigma(\ \mathrm{TF}(\ \mathrm{AGG}(\ \{\boldsymbol{h}_j^{l-1},\ j \in \mathcal{N}(i)\}\ )\ )\ ) \tag{2}$$

where $\boldsymbol{h}_i^l \in \mathbb{R}^{F_l}$ and $\boldsymbol{h}_i^{l-1} \in \mathbb{R}^{F_{l-1}}$ denote the updated and initial node embeddings of node i in layer l, respectively. AGG and TF represent the `aggregation` and `transformation function` which gathers the embeddings of the neighbors and applies a learnable *weight* to the aggregated results of each node, respectively. Different GNNs may differ by their choice of these functions [17, 33]. σ is an activation function that is used to include non-linearity in the model. After the node embeddings are updated by passing through a few GNN layers, a `readout` (also known as pooling) layer may be used to merge the node embeddings into a single graph-level embedding. This embedding will be a vector summarizing the distinctive features of the whole graph.

3 MIS-GNN METHODOLOGY

As mentioned in Sec. 1, our task is to build a GNN model that is faster at predicting the performance of the quantum MIS optimizer. More formally, we define our problem as follows. Let C be an RAA configuration like Fig. 1a. Let $\mathbf{O}$ be either a quantum optimizer that can identify the MIS of C. Let d be a depth value which corresponds to the runtime of $\mathbf{O}$ that would affect $P_{|\mathrm{MIS}|}$ and R. More specifically,

$$\mathbf{O}(C, d) := \Big(P_{|\mathrm{MIS}|}(C, d),\ R(C, d)\Big) \tag{3}$$

We seek to find a prediction function ($\mathbf{F}$) that approximates $P_{|\mathrm{MIS}|}$ and R for any C and d without actually running $\mathbf{O}$:

$$\min_{\mathbf{F}} \left(\operatorname*{avg}_{C,d}\ (\ \mathcal{L}\ (\ \mathbf{O}(C, d),\ \mathbf{F}(C, d)\ )\ ) \right) \tag{4}$$

We define the loss function, $\mathcal{L}$, by taking the root mean squared error (RMSE) of the prediction over the ground-truth value.

To solve this problem, one can analyze the different configurations to detect key features (e.g., grid size, the atom filling ratio, etc.) that result in a difference in the final objectives. Then, a fully-connected network can be employed to learn to predict the final objectives with those features. However, this approach relies on costly and error-prone feature detection. Alternatively, one can treat each of the configurations as an image and apply convolutional neural networks (CNNs) on them. Then, one of this CNN's tasks would be learning to detect the adjacency of the atoms in the configuration. This can be alleviated if we directly input the graph that induced the configuration to the machine learning model. Therefore, exploiting GNNs for building the predictive model is a natural fit for this application. This also helps with reducing the computation intensity of the model. Sec. 3.1 describes our graph representation and Sec. 3.2 explains the architecture of our GNN.

3.1 Graph Representation

As introduced in Sec. 2.1, the unit-disk graph G is specified by the location of its atoms in the corresponding configuration. Specifically, each node is adjacent in G to its 4 nearest neighbors and the 4 next-nearest neighbors if there are indeed atoms at these grid points. For example, in Fig. 1a, the blue node at (3,1) is adjacent to its 3 nearest neighbors (2,1), (3,0), and (4,1), and its 3 next-nearest neighbors (2,2), (4,0), and (4,2).

We choose to build a slightly different graph G' in GNN to benefit the training. Each node in G' keeps their geographical information (X and Y coordinates) as attributes. Edges are directed and have an attribute named *FlowID* to store their type which is one of the eight different directions, as illustrated in Fig. 2a. Considering the example in Fig. 1a, there is an edge from (2,2) to (3,1) with FlowID 7, and also an edge from (3,1) to (2,2) with FlowID 0. From this point forward, when we mention 'graph' or 'G', we are referring to this bi-directional graph.

3.2 Predictive Model

To help MIS-GNN learn better features, we build initial features (embeddings) for each of the nodes and edges based on the attributes we defined in Sec. 3.1. More specifically, the initial node embeddings would include the X and Y coordinates of their respective node as shown in Fig. 2a. Since the depth value has a significant impact on the final objectives, we add it to all the node embeddings of a graph. This enables us to differentiate the MIS instances that have the same configuration but were tested under different depths. As all these features are categorical integer data, we encode them using a one-hot scheme. This is a popular approach for making the machine learning model improve its predictions for such features [34]. The initial node embedding of node i, $\boldsymbol{h}_i^0 \in \mathbb{R}^{F_0}$, is created by concatenating the one-hot encoding of its X coordinate, Y coordinate, and the depth, as depicted in Fig. 2a. Similarly, the edge embedding of the edge from node i to j, $\boldsymbol{e}_{i,j} \in \mathbb{R}^8$, is the one-hot encoding of the FlowID as shown in Fig. 2a.

The first task of the predictive model is to transform the features we defined above based on the graph topology to summarize all the information as a vector $\boldsymbol{h}_G \in \mathbb{R}^{F_G}$. TransformerConv [33] is a state-of-the-art GNN layer that has proven to be highly effective when edges have attributes as well [34]. This is because, in contrast to traditional GNN layers such as GCN [17] where the embeddings of the neighboring nodes are gathered based on predefined weights (e.g., degree of the nodes), TransformerConv learns to identify the importance of a neighboring node. In doing so, it builds an attention coefficient, $\alpha_{i,j}$, for each connection based on the edge embeddings in addition to the source and destination node embeddings. These attention coefficients determine the significance of each neighbor in the aggregation phase. As such, we use TransformerConv as the building block of our GNN encoder as exhibited in Fig. 2b.

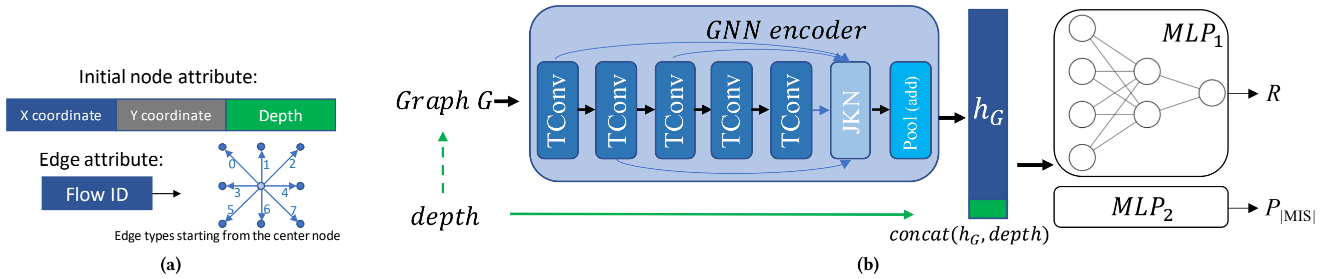

Figure 2: MIS-GNN (a) Initial features of nodes and edges. (b) High-level view of the predictive model.

One known problem with GNNs is over-smoothing, which occurs when the different node embeddings become indistinguishable after a few GNN layers [13]. To alleviate this problem, we exploit a jumping knowledge network (JKN) [41] architecture after the GNN layers as depicted in Fig. 2b, where each node can dynamically determine which layer's output must be used as its final embedding. This can also help the network calibrate a different range of neighborhoods for each of the nodes based on the graph structure. After finalizing the node embeddings, we add them together to create the graph-level embedding, $\boldsymbol{h}_G$.

Once the GNN encoder outputs $\boldsymbol{h}_G$, we have the features of the graph that we are seeking. This brings the network to its second task, which is to predict our final objectives based on these features. To do this transformation, we exploit a multi-layer perceptron (MLP) network as shown in Fig. 2b. As mentioned in Sec. 1, the depth has a high impact on the final result. Therefore, in addition to including it in the initial node features passed to the GNN encoder, we directly pass it to the MLPs as well to amplify its effect on output. For each of the objectives, we define a separate MLP, but they still share the same GNN encoder. This has been shown to increase the accuracy when the objectives are correlated [34].

3.3 Data Augmentation

We employ a data augmentation approach that leverages various transformations of the unit-disk graph, including rotations, mirroring, and diagonal flips. By rotating the graph by 90, 180, and 270 degrees, and reflecting it across vertical, horizontal, and diagonal axes, we generate multiple variants of the original graphs. This process increases the diversity of the dataset, enhancing the robustness of models trained on these augmented samples. The transformations preserve the original graph topology while introducing variations that can help the model generalize better across different spatial configurations.

4 EXPERIMENTAL RESULTS

4.1 Setup and Data Collection

For training and validating our GNN, we collect data with a commercially available RAA, Aquila [29, 40], and also use data from the Harvard study by Ebadi *et al.* [6]. The Harvard data is for both the random instances in Fig. 3 of Ref. [6] and the hard instances for SA considered in Fig 4. However, for consistency we do not include data for the individually optimized instances in Fig. 4, and instead use data for those same graph instances using the same adiabatic algorithm applied to the instances in Fig. 3. The Harvard dataset consists of 130 distinct graphs that include 654 different pairs of depth and graph (d and C in Eq. 3).

Table 1: The statistics of our target dataset. R stands for approximation ratio and $P_{|MIS|}$ denotes the probability of finding an MIS.

source	data	min	max	mean	std
Harvard	depth	1.41	32.0	10.98	10.27
	R	0.795	0.995	0.917	0.043
	$P_{\|MIS\|}$	0.0	0.947	0.173	0.230
	#nodes	20	289	99	81
	#edges	82	1746	559	485
	avg. degree	4.1	6.04	5.42	0.35
Aquila	depth	26.4	26.4	26.4	0.0
	R	0.787	0.946	0.854	0.023
	$P_{\|MIS\|}$	0.0	0.271	0.003	0.02
	#nodes	38	205	125	50
	#edges	75	607	335	145
	avg. degree	4.5	6.09	5.63	0.22

The Aquila dataset consists of another 422 distinct graphs with the same quantum depth. The independent set (IS) results are collected on QuEra Aquila [29, 40] with 1000 shots. We generate atom configurations to realize the unit disk graph by locating the atoms in a grid with spacing of 4um and R_b=6um. We apply the time-dependent Hamiltonian as described in [22] to the system characterizing a graph for solving MIS. Table 1 summarizes the statistics of these graphs and their MIS objectives. We divided the graphs in both datasets into 70%, 15%, and 15% for the training, validation, and testing sets, respectively. The graphs in the testing set are not included in the training set, ensuring they represent unseen data.

The model is deployed and trained using PyTorch [27] with Adam optimizer [16] and a learning rate of 0.005. The final model is picked based on the loss on the validation set and the reported performance numbers are based on the test set. The initial embeddings have 54 features for the annealing/quantum dataset. We use 5 TransformerConv layers with 54 features in the GNN encoder and 4 layers in the MLP decoder. Each GNN layer (except for the last one) is followed by an exponential linear unit (ELU) [5] activation function since it can enable faster training over the traditional rectified linear unit (ReLU) [24] activation function.

4.2 Post-Processing of Experimental Data

The measurement results from Aquila consist of two parts: the `pre_sequence` marks the sites where atom preparation failed, and

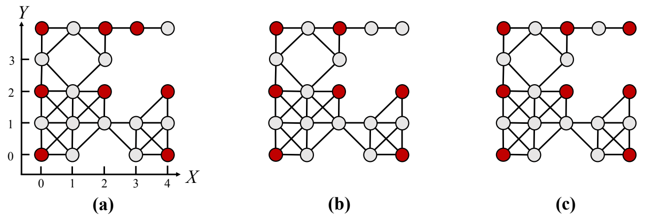

Figure 3: Post-processing of experimental data. (a) After the evolution under the quantum adiabatic algorithm, atoms in the Rydberg state are marked by red dots. The measurement result may not be a valid IS, e.g., atoms at (2,4) and (3,4) are adjacent vertices, so the post-processing is performed to extract a valid IS. (b) The IS after the first phase of the post-processing by removing invalid vertices. (c) The IS after the second phase of the post-processing by including vertices that do not violate the IS constraint.

the `post_sequence` gives the measurement results for the atom state (0 for the Rydberg state, and 1 for the ground state). Then, the IS found by Aquila can be inferred from the measurement outputs by collecting the qubits in the Rydberg state. The raw measurement outputs necessitated postprocessing to ensure the validity of the IS. For example, Figure 3(a) shows raw measure results, where the atoms in the Rydberg stages are marked in red. Those atoms do not constitute a valid IS, since atoms at (2, 4) and (3, 4) cannot be included into the set simultaneously. To transform the measurement outcome to a valid IS, we adopt a two-phase greedy algorithm as described in the paper [6]. In the initial phase, we scan row by row to remove vertices violating the independent set condition. For instance, as shown in Figure 3(b), we begin with checking the atoms with the order: $(0, 0), (3, 0), (0, 2), \ldots, (3, 4)$. If the atom does not share an edge with any atoms in the current IS, we include it to the IS set. In the second phase, the algorithm examines the remaining vertices to reintroduce eligible vertices. According to the row and column order, we iteratively add vertices that can be included in to the IS without violating the constraints. Figure 3(c) demonstrates the final IS after the post-processing. Based on the IS illustrated in Figure 3(b), we follow the order $(1, 0), (2, 0), (0, 1), \ldots, (4, 4)$ to check if a vertex can be added into the IS.

4.3 Model Accuracy Evaluation

As Table 1 shows, the data has a low range. Our experiments show that normalizing the data by a factor of 200 helps the model better distinguish between the objectives, and we can reduce the loss. Table 2 summarizes the loss in the prediction of either of our objectives in their original range. The loss is reported with three different metrics, root mean squared error (RMSE), mean absolute error (MAE), and maximum error. Note that as a lot of $P_{|\text{MIS}|}$ values are very close to zero, mean absolute percentage error (MAPE) would take extreme values, making it inapplicable here. To verify the effectiveness of our optimizations to the model architecture, we compare it to a baseline model that 1) utilizes GCN [17] instead of the TransformerConv, 2) does not include the edge attributes, 3) no jumping knowledge network is used, 4) misses the direct path of the depth value to the input to MLP, 5) employs ReLU activation function instead of ELU, 6) no data augmentation. As the results

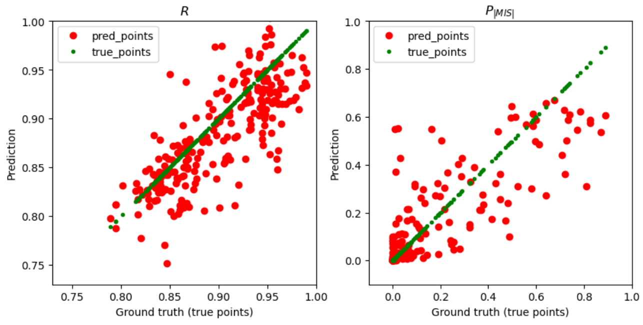

Figure 4: Comparison of the prediction of our model to a perfect model for *quantum experiment*.

Table 2: The model loss in predicting the objectives compared to a baseline model where our optimizations to the network architecture are disabled. Lower is better. MAPE is not applicable since $P_{|\text{MIS}|}$ has many values close to zero.

model	data	RMSE	MAE	max error
baseline	R	0.079	0.057	0.315
	$P_{|\text{MIS}|}$	0.134	0.074	0.611
ours	R	0.033	0.024	0.135
	$P_{|\text{MIS}|}$	0.121	0.065	0.505

demonstrate, our model, due to the optimizations we applied, can get to a lower loss. Compared to the baseline, it can reduce the RMSE by 2.39×, MAE by 2.38×, and max error by 2.3× for predicting approximation ratio. For $P_{|\text{MIS}|}$, we reduce the RMSE by 1.11×, MAE by 1.14×, and max error by 1.21×. The improvement on predicting $P_{|\text{MIS}|}$ is less significant. Since $P_{|\text{MIS}|}$ is biased toward 0 in our dataset, we have limited capability to have accurate prediction for data point with larger $P_{|\text{MIS}|}$.

To better make sense of the model's prediction, we plot their performance against a perfect model. Fig. 4 depicts the results on our dataset. In each plot, we are showing the prediction vs. the ground-truth value. The green points represent a perfect model in which the prediction matches the actual data. The red points depict our model's behavior. Therefore, the closer the red points are to the green points, the better the model's prediction. We can also see that where the density of data is larger (see Table 1), the model can perform better as it has more data to learn from. In consistent with the loss, the model is less accurate when predicting large $P_{|\text{MIS}|}$, since the dataset is unbalance.

5 CONCLUSION & FUTURE WORK

In this paper, we propose a GNN-based predictor, MIS-GNN, for RAA-based quantum MIS optimizer. We convert each graph configuration to a bi-directional graph that encodes the geographical locations of atoms and circuit depth as initial features. We carefully designed a GNN-based model to predict the final objectives by extracting a graph embedding based on the graph topology and the defined features. The experimental results demonstrate that our model can match the ground-truth values with an average RMSE of 0.03 out of the range [0, 1]. Our model is better at predicting the approximation ratio (R) than the success probability in obtaining a MIS ($P_{|\text{MIS}|}$) by the quantum processors. This study is one of the first few attempts (together with [11, 37]) to use machine learning models running on classical computers to predict the performance on quantum processors The error may come from the limited experimental data (e.g. in the GNN-based circuit performance predictor published in [35], over 40,000 circuit configurations were used in training the GNN model), or somehow relate to the quantum uncertainty principle.

We make a large set of RAA experimental data publicly available. These data, along with future data generated by even larger quantum MIS optimizers on the roadmap [30], can contribute to further improvements of the GNN predictor. Furthermore, based on the GNN model presented, it is conceivable to develop generative GNNs that produce instances where quantum advantage emerges, which would help researchers further understand the possibility of quantum advantage in MIS. In doing so, more work is needed to test MIS-GNN on harder instances.

ACKOWLEDGEMENT

This work is funded by NSF grant 2313083. We are grateful for credits provided by Amazon Web Services to use Aquila. We are also grateful to John Long at QuEra for some experimental data. We would also like to thank I. Cong and L. Zhou for helpful discussions at the early phase of this project.

REFERENCES

[1] D. Aharonov and T. Naveh. 2002. Quantum NP - a survey. https://arxiv.org/abs/quant-ph/0210077

[2] Y. Bai, A. Sohrabizadeh, Y. Sun, and J. Cong. 2022. Improving GNN-based accelerator design automation with meta learning. In *2022 Design Automation Conference (DAC)*. 1347–1350.

[3] M. Cain, S. Chattopadhyay, J.-G. Liu, R. Samajdar, H. Pichler, and M. D. Lukin. 2023. Quantum speedup for combinatorial optimization with flat energy landscapes. https://arxiv.org/abs/2306.13123

[4] C.-N. Chou, P. J. Love, J. S. Sandhu, and J. Shi. 2022. Limitations of local quantum algorithms on random Max-k-XOR and beyond. https://arxiv.org/abs/2108.06049

[5] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (ELUs). https://arxiv.org/abs/1511.07289

[6] S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Samajdar, X.-Z. Luo, B. Nash, X. Gao, B. Barak, E. Farhi, S. Sachdev, N. Gemelke, L. Zhou, S. Choi, H. Pichler, S.-T. Wang, M. Greiner, V. Vuletić, and M. D. Lukin. 2022. Quantum optimization of maximum independent set using Rydberg atom arrays. *Science* 376, 6598 (2022), 1209–1215.

[7] E. Farhi, D. Gamarnik, and S. Gutmann. 2020. The quantum approximate optimization algorithm needs to see the whole graph: A typical case. https://arxiv.org/abs/2004.09002

[8] E. Farhi, J. Goldstone, and S. Gutmann. 2014. A quantum approximate optimization algorithm. https://arxiv.org/abs/1411.4028

[9] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser. 2000. Quantum computation by adiabatic evolution. https://arxiv.org/abs/quant-ph/0001106

[10] E. Farhi, J. Goldstone, S. Gutmann, and L. Zhou. 2022. The quantum approximate optimization algorithm and the Sherrington-Kirkpatrick model at infinite size. *Quantum* 6 (July 2022), 759.

[11] D. Fitzek, Y. H. Teoh, H. P. Fung, G. A. Dagnew, E. Merali, M. S. Moss, B. MacLellan, and R. G. Melko. 2024. RydbergGPT. https://arxiv.org/abs/2405.21052

[12] T. Graham, Y. Song, J. Scott, C. Poole, L. Phuttitarn, K. Jooya, P. Eichler, X. Jiang, A. Marra, B. Grinkemeyer, et al. 2022. Multi-qubit entanglement and algorithms on a neutral-atom quantum computer. *Nature* 604, 7906 (2022), 457–462.

[13] Z. Guo, M. Liu, J. Gu, S. Zhang, D. Z. Pan, and Y. Lin. 2022. A timing engine inspired graph neural network model for pre-routing slack prediction. In *2022 Design Automation Conference (DAC)*.

[14] M. P. Harrigan et al. 2021. Quantum approximate optimization of non-planar graph problems on a planar superconducting processor. *Nat. Phys.* 17 (2021), 332.

[15] H. G. Katzgraber, F. Hamze, Z. Zhu, A. J. Ochoa, and H. Munoz-Bauza. 2015. Seeking quantum speedup through spin glasses: The good, the bad, and the ugly. *Phys. Rev. X* 5 (Sep 2015), 15 pages. Issue 3.

[16] D. P. Kingma and J. Ba. 2017. Adam: A method for stochastic optimization. https://arxiv.org/abs/1412.6980

[17] T. N. Kipf and M. Welling. 2017. Semi-supervised classification with graph convolutional networks. https://arxiv.org/abs/1609.02907

[18] J. Kleinberg and E. Tardos. 2006. *Algorithm design*. Pearson Education.

[19] C. Kokail, C. Maier, R. van Bijnen, T. Brydges, M. K. Joshi, P. Jurcevic, C. A. Muschik, P. Silvi, R. Blatt, C. F. Roos, and P. Zoller. 2019. Self-verifying variational quantum simulation of lattice models. *Nature* 569, 7756 (01 May 2019), 355–360.

[20] M. Kou, J. Zeng, B. Han, F. Xu, J. Gu, and H. Yao. 2022. GEML: GNN-based efficient mapping method for large loop applications on CGRA. In *2022 Design Automation Conference (DAC)*. 337–342.

[21] J.-G. Liu, X. Gao, M. Cain, M. D. Lukin, and S.-T. Wang. 2023. Computing solution space properties of combinatorial optimization problems via generic tensor networks. *SIAM Journal on Scientific Computing* 45, 3 (2023), A1239–A1270.

[22] P. Lopes, A. Keesling, M. Lin, and P. Komar. 2022. *Optimization with a Rydberg atom-based quantum processor*. https://aws.amazon.com/blogs/quantum-computing/optimization-with-rydberg-atom-based-quantum-processor/

[23] M. D. Lukin, M. Fleischhauer, R. Cote, L. M. Duan, D. Jaksch, J. I. Cirac, and P. Zoller. 2001. Dipole blockade and quantum information processing in mesoscopic atomic ensembles. *Phys. Rev. Lett.* 87 (2001), 4 pages. Issue 3.

[24] V. Nair and G. E. Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *2010 International Conference on International Conference on Machine Learning (ICML)*.

[25] M.-T. Nguyen, J.-G. Liu, J. Wurtz, M. D. Lukin, S.-T. Wang, and H. Pichler. 2023. Quantum optimization with arbitrary connectivity using Rydberg atom arrays. *PRX Quantum* 4, 1 (2023), 010316.

[26] G. Pagano, A. Bapat, P. Becker, K. S. Collins, A. De, P. W. Hess, H. B. Kaplan, A. Kyprianidis, W. L. Tan, C. Baldwin, L. T. Brady, A. Deshpande, F. Liu, S. Jordan, A. V. Gorshkov, and C. Monroe. 2020. Quantum approximate optimization of the long-range Ising model with a trapped-ion quantum simulator. *Proc. of the Ntl. Acad. Sci.* 117, 41 (Oct. 2020), 25396–25401.

[27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In *2019 International Conference on Neural Information Processing Systems (NeurIPS)*.

[28] H. Pichler, S.-T. Wang, L. Zhou, S. Choi, and M. D. Lukin. 2018. Quantum optimization for maximum independent set using Rydberg atom arrays. https://arxiv.org/abs/1808.10816

[29] QuEra. 2022. Meet Aquila, QuEra's 256-qubit quantum processor. https://www.quera.com/aquila

[30] QuEra. 2022. Our quantum roadmap. https://www.quera.com/our-quantum-roadmap

[31] H. Ren, S. Nath, Y. Zhang, H. Chen, and M. Liu. 2022. Why are graph neural networks effective for EDA problems?. In *2022 International Conference on Computer Aided Design (ICCAD)*.

[32] T. F. Rønnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, and M. Troyer. 2014. Defining and detecting quantum speedup. *Science* 345, 6195 (2014), 420–424.

[33] Y. Shi, Z. Huang, W. Wang, H. Zhong, S. Feng, and Y. Sun. 2021. Masked label prediction: unified message passing model for semi-supervised classification. *2021 International Joint Conference on Artificial Intelligence (IJCAI)* (2021).

[34] A. Sohrabizadeh, Y. Bai, Y. Sun, and J. Cong. 2022. Automated accelerator optimization aided by graph neural networks. In *2022 Design Automation Conference (DAC)*.

[35] A. Sohrabizadeh, Y. Bai, Y. Sun, and J. Cong. 2023. Robust GNN-based representation learning for HLS. In *2023 International Conference on Computer Aided Design (ICCAD)*.

[36] Q. Tan, N. Liu, and X. Hu. 2019. Deep representation learning for social network analysis. *Frontiers in Big Data* 2 (2019), 2.

[37] H. Wang, M. Weber, J. Izaac, and C. Y.-Y. Lin. 2024. Predicting properties of quantum systems with conditional generative models. https://arxiv.org/abs/2211.16943

[38] D. Wecker, M. B. Hastings, and M. Troyer. 2016. Training a quantum optimizer. *Phys. Rev. A* 94 (Aug 2016), 022309. Issue 2.

[39] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. 2021. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2021), 4–24.

[40] J. Wurtz, A. Bylinskii, B. Braverman, J. Amato-Grill, S. H. Cantu, F. Huber, A. Lukin, F. Liu, P. Weinberg, J. Long, S.-T. Wang, N. Gemelke, and A. Keesling. 2023. Aquila: QuEra's 256-qubit neutral-atom quantum computer. https://arxiv.org/abs/2306.11727

[41] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *2018 International Conference on International Conference on Machine Learning (ICML)*. 5453–5462.

[42] A. P. Young, S. Knysh, and V. N. Smelyanskiy. 2010. First-order phase transition in the quantum adiabatic algorithm. *Phys. Rev. Lett.* 104 (Jan 2010), 020502. Issue 2.