Meta-Learning for Quantum Optimization via Quantum Sequence Model

Yu-Cheng Lin *, Yu-Chao Hsu†‡¶, Samuel Yen-Chi Chen§‖,
* Department of Electrophysics, National Yang Ming Chiao Tung University, Hsinchu, Taiwan
† National Center for High-Performance Computing, National Institutes of Applied Research, Hsinchu, Taiwan
‡ Cross College Elite Program, National Cheng Kung University, Tainan, Taiwan
§ Wells Fargo, New York, NY, USA

***Abstract*—The Quantum Approximate Optimization Algorithm (QAOA) is a leading approach for solving combinatorial optimization problems on near-term quantum processors. However, finding good variational parameters remains a significant challenge due to the non-convex energy landscape, often resulting in slow convergence and poor solution quality. In this work, we propose a quantum meta-learning framework that trains advanced quantum sequence models to generate effective parameter initialization policies. We investigate four classical or quantum sequence models, including the Quantum Kernel-based Long Short-Term Memory (QK-LSTM), as learned optimizers in a "learning to learn" paradigm. Our numerical experiments on the Max-Cut problem demonstrate that the QK-LSTM optimizer achieves superior performance, obtaining the highest approximation ratios and exhibiting the fastest convergence rate across all tested problem sizes ($n = 10$ to 13). Crucially, the QK-LSTM model achieves perfect parameter transferability by synthesizing a single, fixed set of near-optimal parameters, leading to a remarkable sustained acceleration of convergence even when generalizing to larger problems. This capability, enabled by the compact and expressive power of the quantum kernel architecture, underscores its effectiveness. The QK-LSTM, with only 43 trainable parameters, substantially outperforms the classical LSTM (56 parameters) and other quantum sequence models, establishing a robust pathway toward highly efficient parameter initialization for variational quantum algorithms in the NISQ era.**

***Index Terms*—Meta-Learning, Quantum Optimization, Quantum Machine Learning, Quantum Kernel Method.**

I. Introduction

Optimization problems are widely applied across fields such as finance [1], logistics [2], and healthcare [3]. Despite significant progress, classical optimization algorithms face limitations in computational resources and algorithmic constraints when dealing with large-scale or complex problems, preventing efficient handling.

To overcome these challenges, quantum optimization methods, particularly the Quantum Approximate Optimization Algorithm (QAOA) [4] is proposed to solve combinatorial optimization problems and other NP-hard tasks [5] using shallow quantum circuits on near-term quantum devices [6]–[8].

¶ an4114752@gs.ncku.edu.tw. ‖ ycchen1989@ieee.org.

Despite its potential advantages, optimizing the parameters of QAOA remains a significant challenge. The optimization landscape is inherently non-convex, characterized by multiple local minimal [9], while the performance is further affected by hardware noise and measurement uncertainties present in near-term quantum devices. From the viewpoint of classical optimization, Bittel et al. [10] showed that, even if a quantum system can be efficiently simulated classically, the corresponding classical optimization process still remains NP-hard.

Moreover, the presence of barren plateaus [11]–[13] and narrow gorges [14] during training significantly complicates the optimization process, often preventing the model from achieving stable convergence. To address this issue, meta-learning [15], [16] has emerged as a promising paradigm that enables models to learn how to optimize by leveraging prior experience across multiple related tasks.

The concept of a learned optimizer was first introduced by Andrychowicz et al. [17], where a coordinate-wise Long Short-Term Memory (LSTM) network was used to replace traditional hand-crafted optimizers.

In this work, we propose a quantum sequence model that integrates meta-learning with quantum-enhanced learning-based optimizers to substantially improve the efficiency and performance of quantum optimization algorithms. Specifically, we develop three meta-learned architectures within this framework such as Quantum Long Short-Term Memory (QLSTM) [18], [19], Quantum Kernel-based LSTM (QK-LSTM) [20]–[22], and Quantum Fast Weight Programmer (QFWP) [23]–[25], each designed to capture different aspects of temporal dynamics and optimization behavior in quantum systems. We show that in Section II.

In this way our approach demonstrates significant improvements in convergence speed and solution quality. In particular, QK-LSTM and QLSTM outperform conventional classical optimization methods across multiple benchmark tests. Furthermore, we conduct an extensive benchmarking study, systematically comparing these quantum meta-learning architectures with various classical optimizers to comprehensively evaluate their performance and robustness. The experimental results are presented and discussed in Section IV. Finally, we discuss and conclude the paper in Section V and VI.

II. Quantum Sequence Model

A. Classical Long Short-Term Memory

The LSTM networks (see Fig. 1) have profoundly influenced the development of machine learning and sequential data modeling [26]. By effectively addressing the vanishing and exploding gradient problems that limit the Recurrent Neural Network (RNN) [27], it offers a stable mechanism for learning long-term dependencies in sequential data. LSTM networks have been widely applied in various fields, including time series forecasting [28]–[30], natural language processing [31]–[34], and medical applications [35]–[37]. In this work, we employ LSTM networks for meta-learning [38].

$$\mathbf{f}_t = \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f), \tag{1a}$$
$$\mathbf{i}_t = \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i), \tag{1b}$$
$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c), \tag{1c}$$
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t, \tag{1d}$$
$$\mathbf{o}_t = \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o), \tag{1e}$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{1f}$$

where:

- $x_t \in \mathbb{R}^n$ is the input vector at time t,
- $h_{t-1} \in \mathbb{R}^m$ is the hidden state from the previous time step,
- W_f, W_i, W_C, W_o are weight matrices,
- b_f, b_i, b_C, b_o are bias vectors,
- $\sigma(\cdot)$ denotes the sigmoid activation function,
- $\tanh(\cdot)$ denotes the hyperbolic tangent activation function,
- $\odot$ represents element-wise multiplication.

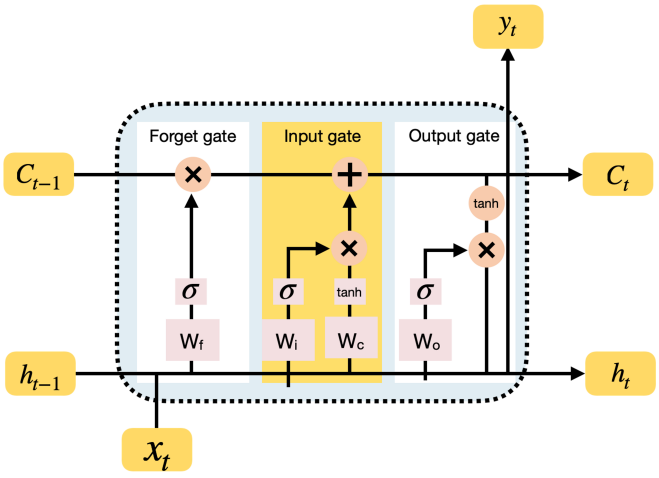

Fig. 1: Schematic representation of the LSTM.

B. Quantum Long Short-Term Memory

The QLSTM model [18] extends the classical LSTM by replacing its classical neural network components with variational quantum circuit (VQC) [39]–[41], thereby exploiting quantum mechanical properties such as superposition and entanglement (see Fig. 2 and Fig. 3). The core operation of the QLSTM lies in its gating mechanism, which regulates the flow of information through quantum-encoded states.

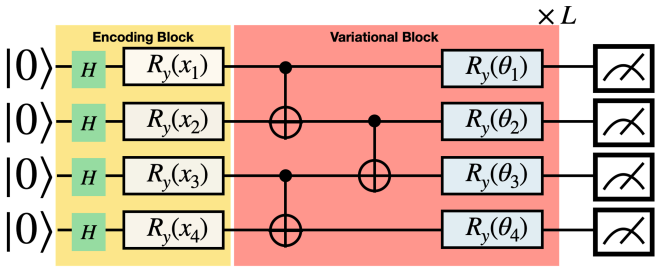

Fig. 2: Schematic representation of the VQC.

$$\mathbf{f}_t = \sigma(\text{VQC}_1(v_t)), \tag{2a}$$
$$\mathbf{i}_t = \sigma(\text{VQC}_2(v_t)), \tag{2b}$$
$$\tilde{\mathbf{c}}_t = \tanh(\text{VQC}_3(v_t)), \tag{2c}$$
$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t, \tag{2d}$$
$$\mathbf{o}_t = \sigma(\text{VQC}_4(v_t)), \tag{2e}$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \tag{2f}$$
$$\tag{2g}$$

where $v_t = [h_{t-1}, x_t]$ denotes the concatenation of the current input x_t at time step t and the hidden state h_{t-1} from the previous time step.

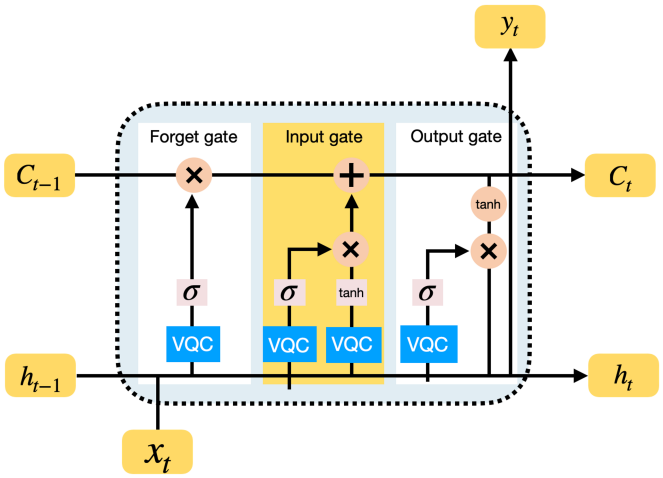

Fig. 3: Schematic representation of the QLSTM.

C. Quantum Kernel-Based Long Short-Term Memory

The QK-LSTM Network [20]–[22], a hybrid machine learning architecture that combines the strengths of classical LSTM networks with quantum kernel method [42]–[44] (see Fig. 4). This model is employed to enhance performance and reduce the number of trainable parameters in meta-learning tasks.

The formal mathematical formulation of the QK-LSTM cell is defined as follows:

$$f_t = \sigma\left(\sum_{j=1}^{N} \beta_j^{(f)}\, \kappa^{(f)}(v_t, v_j)\right), \tag{3a}$$

$$i_t = \sigma\left(\sum_{j=1}^{N} \beta_j^{(i)}\, \kappa^{(i)}(v_t, v_j)\right), \tag{3b}$$

$$\hat{C}_t = \tanh\left(\sum_{j=1}^{N} \beta_j^{(C)}\, \kappa^{(C)}(v_t, v_j)\right), \tag{3c}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t, \tag{3d}$$

$$o_t = \sigma\left(\sum_{j=1}^{N} \beta_j^{(o)}\, \kappa^{(o)}(v_t, v_j)\right), \tag{3e}$$

$$h_t = o_t \odot \tanh\left(C_t\right), \tag{3f}$$

where:

- $v_t = [h_{t-1}; x_t] \in \mathbb{R}^{n+m}$ represents the concatenation of input x_t at time-step t and the hidden state h_{t-1} from the previous time-step $t-1$,
- $\beta_j^{(f)}$, $\beta_j^{(i)}$, $\beta_j^{(C)}$, and $\beta_j^{(o)}$ are trainable coefficients corresponding to each gate's quantum kernel,
- $\kappa^{(f)}(\cdot,\cdot)$, $\kappa^{(i)}(\cdot,\cdot)$, $\kappa^{(C)}(\cdot,\cdot)$, and $\kappa^{(o)}(\cdot,\cdot)$ denote the quantum kernel functions tailored to the respective gates.

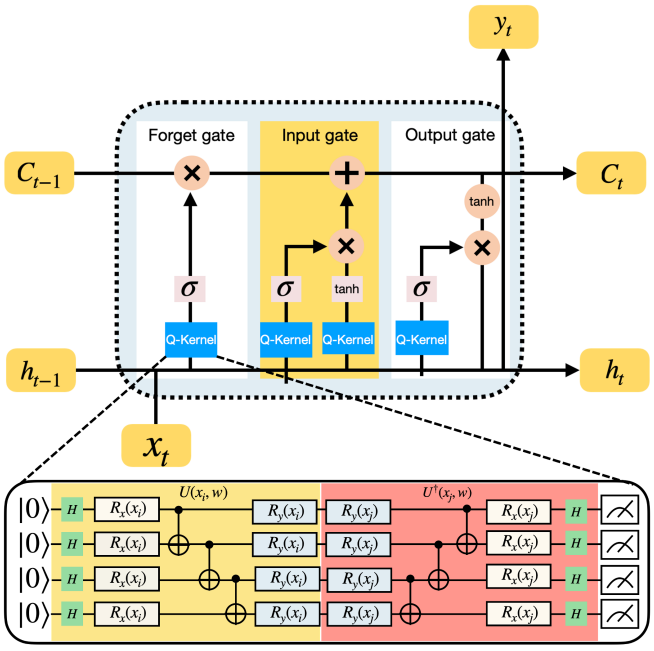

Fig. 4: Schematic illustration of the QK-LSTM and the quantum circuit representation of $U(x_i, w)$ and $U^\dagger(x_j, w)$, demonstrating the use of quantum gates for encoding data and extracting features for machine learning applications.

D. Quantum FWP

The classical Fast Weight Programmer (FWP) model [45] consists of two neural networks: a *slow network* and a *fast network*, where the weights of the fast network act as the program. The slow network updates the weights of the fast network at each time step based on incoming data, without completely overwriting them. This mechanism enables the model to perform rapid short-term adaptation while preserving long-term information.In this way, it eliminates the need for recurrent connections, thereby reducing computational overhead.

Inspired by this architecture, the Quantum Fast Weight Programmer (QFWP) (see Fig. 5) [23] replaces the classical neural networks of both the slow and fast programs with VQC. The classical input vector $\vec{x}$ is mapped to latent vectors $\mathbf{L} \in \mathbb{R}^L$ and $\mathbf{Q} \in \mathbb{R}^n$.

The outer product $\mathbf{L} \times \mathbf{Q}$ is then computed to generate an update term for the VQC parameters. At each time step $t+1$, the parameters of the quantum circuit are updated according to $\theta_{ij}^{t+1} = f(\theta_{ij}^t, \mathbf{L}_i \times \mathbf{Q}_j)$, where $f(\cdot)$ combines the parameters from the previous time step θ_{ij}^t with the newly generated term $\mathbf{L}_i \times \mathbf{Q}_j$. This approach enables the circuit parameters to retain information from previous time steps. The output of the VQC can be further refined through post-processing layers, such as normalization, translation, or classical neural networks, to enhance the final results.

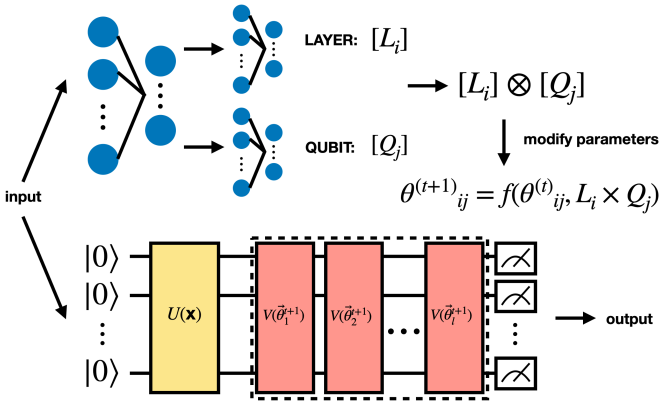

Fig. 5: Schematic representation of the QFWP.

III. Method

All numerical simulations and model training processes were performed using Python (v3.10.18).The primary libraries utilized include PennyLane (v0.42.3) [46] for quantum circuit simulation, PyTorch (v2.8.0) [47] for neural network implementation and training, and SciPy (v1.15.3) [48] for numerical operation. All experiments were executed on a system equipped with eight NVIDIA Tesla V100 GPUs (16GB) and Intel Xeon CPUs.

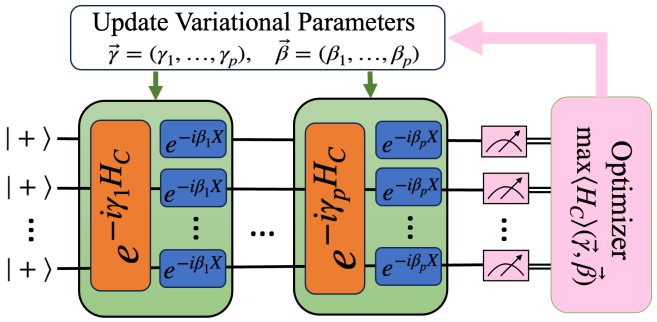

Fig. 6: Schematic representation of the QAOA.

A. Quantum Approximate Optimization Algorithms for Max-Cut

This work addresses the optimization challenge of the QAOA (see Fig. 6) applied to the Max-Cut problem on an unweighted graph $G \in (V, E)$ [4]. The max-cut problem is formulated as identifying a vertex partition that maximizes the number of edges crossing the cut.

The QAOA ansatz acts on the initial uniform superposition state $|\psi_0\rangle = |+\rangle^{\otimes n}$, where $n = |V|$ is the number of qubits. The variational state is then defined by the alternating application of the cost and mixer Hamiltonians:

$$|\psi(\vec{\theta})\rangle = \prod_{j=1}^{P} e^{-i\beta_j \hat{H}_M} e^{-i\gamma_j \hat{H}_C} |\psi_0\rangle \tag{4}$$

where $\vec{\theta} = (\vec{\beta}, \vec{\gamma}) \in \mathbb{R}^{2P}$, and $\vec{\beta}$ and $\vec{\gamma}$ is the vector of variational parameters, and P is the number of layers and the cost Hamiltonian $\hat{H}_C$ encodes the Max-Cut objective via a sum of pairwise $\hat{Z}$ operators:

$$\hat{H}_C = \sum_{\{j,k\} \in e} \frac{1}{2}(\hat{I} - \hat{Z}_j \hat{Z}_k) \tag{5}$$

In this study, the QAOA layer is fixed at $P = 1$, yielding a total of 2 variational parameters, $\vec{\theta} = (\beta_1, \gamma_1)$.

The objective function to be maximized is the expected cut value:

$$f(\vec{\theta}) = \langle \psi(\vec{\beta}, \vec{\gamma}) | \hat{H}_C | \psi(\vec{\beta}, \vec{\gamma}) \rangle \tag{6}$$

We generate random problem instances using the Erdős–Rényi graph model. For training, we sampled 1008 graphs with $n \in [6, 9]$ nodes. For testing, 90 graphs with $n \in [10, 13]$ nodes are generated. In both cases, the connectivity probability of each edge is $p = k/n$, where $k \in [3, n-1]$.

B. Meta-Learning Framework for QAOA

The main challenge in optimizing the QAOA lies in the inherent inefficiency and susceptibility to local minima within the classical optimization loop. To overcome this, we use the parameter tuning process as a sequence prediction problem within the meta-learning paradigm, commonly referred to as "learning to learn" (L2L) [17].

Taking inspiration from the pioneering work of Verdon *et al.* [49], which utilized RNN for quantum parameter initialization (see Fig. 7), we extend their approach to a comparative study encompassing a suite of advanced quantum and classical sequence models: QK-LSTM, QLSTM, and QFWP. In this meta-learning context, the objective is to train an optimization agent (Model_ϕ) to generate an effective update policy for the QAOA variational parameters ($\vec{\theta}$) based on the historical trajectory of the optimization. The underlying intuition for this approach is that the variational parameters (γ and β) are analogous to hyperparameters governing descent dynamics in the rugged cost landscape [4]. A sequence model, trained on diverse small problem instances, learns a generalized, problem-agnostic update that effectively approximates the near-optimal control sequence, thereby significantly reducing the quantum-classical iterations required for initialization.

For consistency and fair comparison across the models investigated, structural adjustments were implemented to match the output dimension of the QAOA parameters. Four qubits were used for the internal quantum operations within the QLSTM and QK-LSTM models. Since the QAOA parameter dimension of 4, a fully connected (FC) layer was appended downstream to the QK-LSTM and QLSTM models to reshape their output to the required parameter size, while the standard LSTM and QFWP models were structured to output the final parameter size directly.

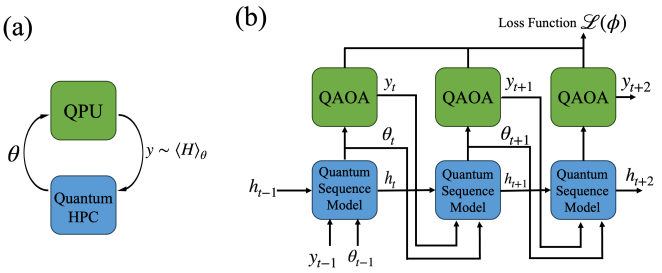

Fig. 7: Schematic representation of the QAOA framework with Meta Learning. (a) Quantum processing unit (QPU) interacting with Quantum HPC, where the quantum state θ is evaluated with the Hamiltonian $\langle H \rangle_\theta$. (b) Architecture of a quantum sequence model using QAOA blocks for optimizing time-series predictions.

C. Optimization Protocol

Our optimization strategy consists of a meta-learning process to train the sequence models, followed by a two-phase optimization protocol during testing to evaluate performance.

1) Model Training (Meta-Learning): The models were trained using small graphs $n \in [6, 9]$ to learn an effective parameter update policy ϕ. The training process utilized a fixed time horizon of $T = 10$ steps. The model functions as a meta-optimizer, which use the previous state h_t, the parameter vector $\vec{\theta}_t$, and the estimated cost y_t at each step t to predict the next parameter set:

$$h_{t+1}, \vec{\theta}_{t+1} = \text{Model}_\phi(h_t, \vec{\theta}_t, y_t) \tag{7}$$

The initial parameters for the training loop were set to $\vec{\theta}_0 = (0^{\otimes 2P})$. The model parameters ϕ were optimized using RMSprop via backpropagation through time [50] for a maximum of 50 epochs.

The meta-loss $\mathcal{L}(\phi)$ was computed as a weighted sum of the QAOA costs $f(\vec{\theta})$ across 10 steps:

TABLE I: COMPANISON OF PARAMETERS FOR SEQUENCE MODELS

Parameter	QK-LSTM	LSTM	QLSTM	QFWP
Epochs	50	50	50	50
Recurrent Steps (T)	10	10	10	10
Learning Rate (Sequence Model)	6×10^{-6}	6×10^{-6}	6×10^{-6}	6×10^{-6}
Learning Rate (FC Layer)	1×10^{-4}	-	1×10^{-4}	-
Number of Qubits in Circuit	4	-	4	2
Total Trainable Parameters	**43**	**56**	**43**	**31**

TABLE II: Average NUMBER of iterations to CONVERGENCE (INCLUDE $T = 10$ RECURRENT STEPS FOR SEQUENCE MODELS) FOR Node 10 THROUGH 13

num_node	QK-LSTM	LSTM	QLSTM	QFWP	Random seed
10	**48**	86	57	128	177
11	**45**	82	49	141	155
12	**41**	76	49	111	146
13	**35**	72	47	177	145

$$\mathcal{L}(\phi) = \mathbb{E}[\sum_{t=1}^{T} 0.1t \cdot f(\vec{\theta_t})] \tag{8}$$

All models were trained with identical sequence model hyperparameters for rigorous comparison, as detailed in Table I.

2) Test Optimization Protocol: For testing on larger, unseen graphs $n \in [10, 13]$, a two-phase optimization protocol was executed. In the first phase (Phase I), the trained sequence model was run for 10 steps starting from $\vec{\theta_0}$ to generate a high-quality initial parameter set $\vec{\theta}_{10}$. In Phase II, the parameter set $\vec{\theta}_{10}$ from Phase I was used as the seed for fine-tuning. This optimization continued using the classical Stochastic Gradient Descent (SGD) [51] optimizer with a fixed learning rate of 1×10^{-3}.

To comparison with standard QAOA, we used the same SGD optimizer with random seed.

IV. Results

To quantify the efficacy of the meta-learning initialization approach, we evaluated the performance of the sequence model-QAOA frameworks on 90 unseen Max-Cut graphs ($n = 10$ to $n = 13$). The overall performance was benchmarked against the standard random-seed QAOA baseline.

Figure 8 displays the average relative error $\overline{f}_{\text{rel}}(j)$ as a function of the total optimization steps for Max-Cut instances with $n = 10$ through $n = 13$. The dashed curves graphically illustrate the optimization trajectory from the sequence model's recurrent steps (Phase I), while the subsequent solid lines represent the optimization path in Phase II, which was described in Section III. The vertical dashed lines indicate the average iteration to convergence for each model, which is further detailed below.

The relative error $f_{\text{rel}}(\vec{\theta})$ measures the gap between the expected cost value $f(\vec{\theta})$ and the globally optimal cost value $f_{\min}$, which was obtained via brute-force search.

$$f_{\text{rel}}(\vec{\theta}) = \frac{f(\vec{\theta}) - f_{\min}}{f_{\min}} \tag{9}$$

The reported performance is based on the average relative error $\overline{f}_{\text{rel}(j)}$, which is defined as the average of the relative error at optimization step j across all testing graph instances (N_{test}) for a given nodes n:

$$\overline{f}_{\text{rel}}(j) = \frac{1}{N_{\text{test}}} \sum_{k=1}^{N_{\text{test}}} f_{\text{rel}}^{(k)}(j) \tag{10}$$

Table II quantified the average number of iterations required for convergence for each sequence model-QAOA framework and the random-seed QAOA baseline across varying node sizes. The average iterations to convergence is defined as the total number of optimization steps j (including the initial $T = 10$ recurrent steps) required for the optimization trajectory to stabilize. Specifically, convergence is considered reached when the absolute step-to-step change in the average relative error falls below a tolerance of $\epsilon = 10^{-4}$.

$$|\overline{f}_{\text{rel}}(j+1) - \overline{f}_{\text{rel}}(j)| \leq \epsilon \tag{11}$$

Table III details the approximation ratio [9] (mean $\pm$ std) achieved at the end of the two phases described in Section III. The approximation ratio is defined as the ratio of the expected cut value $C(z)$ to the optimal cut value $C(z^*)$

$$\text{Approx. Ratio}(z) = \frac{C(z)}{C(z^*)} \tag{12}$$

with the cut value $C(z)$ defined as:

$$C(z) = \frac{1}{2}(\sum_{(i,j)\in E} w_{ij} - \sum_{(i,j)\in E} w_{ij} z_i z_j) \tag{13}$$

This metric, bounded by $(0, 1]$, is essential for assessing the quality of the final combinatorial solution.

V. Discussion

As demonstrated in Figure 8, the QK-LSTM model achieved the highest quality approximate optimum of the QAOA parameters within the fixed 10 iteration compared to the other three sequence models and the standard QAOA with a random seed baseline. The results clearly show that the QK-LSTM framework provides the fastest and most stable convergence trajectory among all sequence models.

This superior convergence translates directly into higher quality solutions. Table III reports the approximation ratios (mean $\pm$ std) at iteration 10. The QK-LSTM consistently exhibits the highest approximation ratio across various node sizes, underscoring its resilience and capacity to swiftly navigate the cost landscape to locate high quality solutions. While

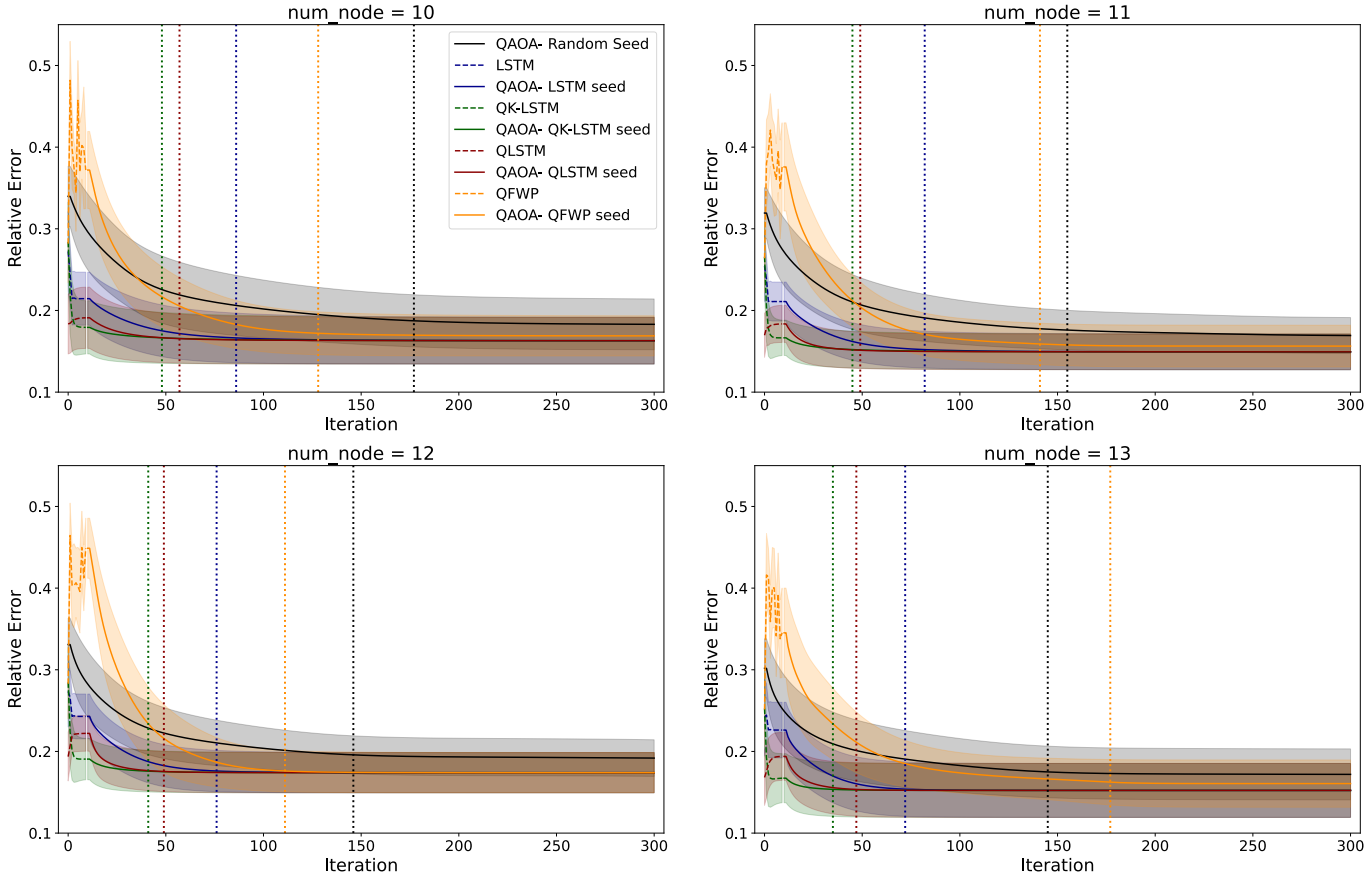

Fig. 8: Average relative errors over 300 iterations for sequence models on MaxCut problems with nodes $n = 10$ to 13. Dashed lines denote sequence model optimization; solid lines show post-seeding SGD optimization. Vertical lines indicate the average iteration to convergence (detailed in Table II). Error bars represent the 95% confidence interval.

TABLE III: APPROXIMATION RATIO (MEAN $\pm$ STD) ACHIEVED BY THESEQUENCE MOODEL (PHASE I, $T = 10$) AND THE SUBSEQUENT QAOA WITH SEQUENCE MODEL SEEDING (PHASE II) AT ITERATION 10 FOR NODE 10 THROUGH 13. ALL VALUE ARE ROUNDED TO TWO DECIMALS.

num_node	**Phase I**				**Phase II**				
	QK-LSTM	**LSTM**	**QLSTM**	**QFWP**	**QK-LSTM**	**LSTM**	**QLSTM**	**QFWP**	**Random seed**
10	**0.82 ± 0.03**	0.79 ± 0.03	0.81 ± 0.04	0.63 ± 0.05	**0.83 ± 0.03**	0.80 ± 0.03	0.82 ± 0.03	0.70 ± 0.05	0.70 ± 0.05
11	**0.83 ± 0.01**	0.79 ± 0.02	0.82 ± 0.02	0.62 ± 0.05	**0.84 ± 0.01**	0.81 ± 0.02	0.84 ± 0.02	0.70 ± 0.05	0.72 ± 0.04
12	**0.81 ± 0.03**	0.76 ± 0.03	0.78 ± 0.02	0.55 ± 0.04	**0.82 ± 0.03**	0.78 ± 0.03	0.81 ± 0.03	0.66 ± 0.05	0.71 ± 0.04
13	**0.83 ± 0.03**	0.77 ± 0.03	0.81 ± 0.03	0.65 ± 0.05	**0.84 ± 0.03**	0.80 ± 0.03	0.83 ± 0.03	0.72 ± 0.05	0.75 ± 0.04

both QLSTM and LSTM also outperformed standard random initialization, their approximation ratios were notably lower than the quantum kernel enhanced model, confirming the specific advantage of the hybrid QK-LSTM architecture.

The success of the sequence models in this optimization task is rooted in their ability to quickly identify and propose initialization parameters that lie within a favorable basin of attraction on the QAOA cost landscape (see Fig. 9). Figure 9 serves as an illustrative example of this mechanism, and consistent with this observation, parameter sets initialized by the QK-LSTM consistently led to the highest quality optima across all tested node sizes when compared to initializations derived from QLSTM, LSTM, QFWP, and the standard QAOA with a random seed. This finding validates the core hypothesis: the sequence models, particularly QK-LSTM, serve as highly effective meta-models capable of learning the optimal optimization trajectory from historical data.

The QK-LSTM model demonstrates significant strides in parameter efficiency by effectively incorporating the quantum kernel function within a classical LSTM architecture [20]. This integration not only leverages quantum feature spaces to capture intricate data dependencies but also achieves model compression with fewer trainable parameters than the LSTM model, all without sacrificing accuracy. This efficiency

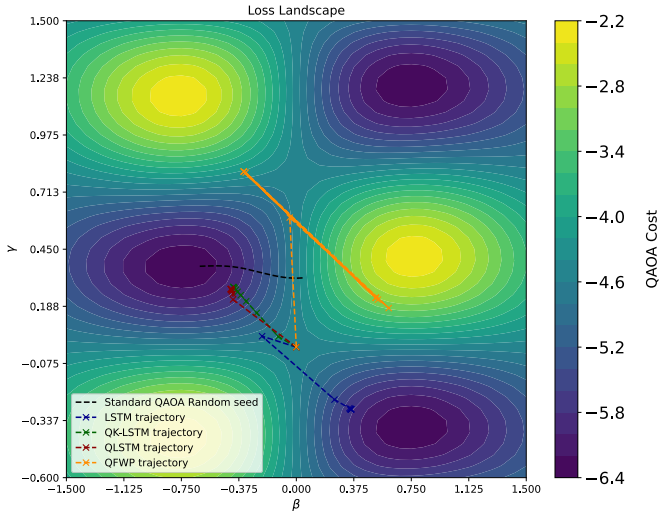

Fig. 9: Comparison of four sequence models and standard QAOA with random seed optimization trajectories in the single-layer QAOA parameter space, evaluated on a randomly generated graph with 10 nodes and edge probability $p = 4/10$. All sequence models start from the same initial point $(0, 0)$.

makes the QK-LSTM particularly suitable for deployment in resource-constrained environments.

An interesting observation is that as the number of nodes increases, the average iteration count required for convergence of QK-LSTM, QLSTM, and LSTM decreases. This suggests that the meta-models are highly effective and potentially even better suited for the large-sclae problem sizes typical of challenge quantum optimization tasks, indicating that the predictive signal in the training data may become clearer with increasing graph complexity.

It is important to note that the performance of our LSTM meta-model was lower than that observed in the prior literature [49]. This can be attributed to two main factors related to our experimental setup, which was fixed across all four models for a fair comparison: the use of a single, uniform learning rate for all models and our choice of a simpler meta-loss function, as opposed to the *observed improvement* metric proposed in previous work [49].

VI. Conclusion and Future Work

In this paper, we extended the application of meta-learning to QAOA optimization by proposing a novel comparison between four sequence models— QK-LSTM, QLSTM, LSTM, FWP, and standard QAOA—to solve the Max-Cut problem.

Our numerical experiments highlight the QK-LSTM's superior performance. It demonstrated a remarkable ability to converge rapidly to near-optimal solutions, significantly outperforming the other three sequence models and the standard QAOA with random seed in terms of both iteration count and approximation ratios. We established that these models are used to quickly find a high quality global approximate optimum of the parameters, which then serves as a powerful initialization point for subsequent local search heuristic. This combined approach yielded high quality optima in the QAOA cost landscape, requiring substantially fewer quantum-classical optimization iterations than the traditional alternatives. Furthermore, the QK-LSTM's transfer-learning capability was shown to allow successful deployment on larger instances after training on smaller graphs, significantly reducing overall runtime and computational overhead.

In terms of possible extensions of this work, the meta-learning approach holds promise for application in several complex domains, including the Sherrington-Kirkpatrick Ising spin glass [52] and variational quantum eigensolver ansatz for preparing ground states of Hubbard models [53] and molecular systems [54]. Future studies should also focus on testing the performance of these models under various hardware noise conditions. We envision that these future investigations will refine the quantum meta-learning paradigm, accelerating the practical and efficient deployment of quantum optimization algorithms in real-world realms such as materials science, logistics, and finance.

Code Availability

All code and data used in this study are available at the following GitHub repository: https://github.com/Astor-Hsu/QAOA-Meta-Learning.

Acknowledgment

The authors would like to thank the National Center for High-performance Computing of Taiwan for providing computational and storage resources.

References

[1] A. S. Naik, E. Yeniaras, G. Hellstern, G. Prasad, and S. K. L. P. Vishwakarma, "From portfolio optimization to quantum blockchain and security: A systematic review of quantum computing in finance," *Financial Innovation*, vol. 11, no. 1, pp. 1–67, 2025.

[2] G. Rahmanifar, "Green vehicle routing and logistics optimization: Iot-driven solutions for modern urban challenges," 2025.

[3] E. Cabrera *et al.*, "Simulation optimization for healthcare emergency departments," *Procedia computer science*, vol. 9, pp. 1464–1473, 2012.

[4] E. Farhi *et al.*, "A quantum approximate optimization algorithm," *arXiv preprint arXiv:1411.4028*, 2014.

[5] Y. Chatterjee *et al.*, "Solving various np-hard problems using exponentially fewer qubits on a quantum computer," *Physical Review A*, vol. 109, no. 5, p. 052441, 2024.

[6] L. Zhou *et al.*, "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices," *Physical Review X*, vol. 10, no. 2, p. 021067, 2020.

[7] P. C. Lotshaw *et al.*, "Scaling quantum approximate optimization on near-term hardware," *Scientific Reports*, vol. 12, no. 1, p. 12388, 2022.

[8] K.-C. Chen *et al.*, "Noise-aware distributed quantum approximate optimization algorithm on near-term quantum hardware," in *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 2, pp. 144–149, IEEE, 2024.

[9] K.-C. Chen *et al.*, "Learning to learn with quantum optimization via quantum neural networks," *arXiv preprint arXiv:2505.00561*, 2025.

[10] L. Bittel and M. Kliesch, "Training variational quantum algorithms is np-hard," *Physical review letters*, vol. 127, no. 12, p. 120502, 2021.

[11] J. R. McClean *et al.*, "Barren plateaus in quantum neural network training landscapes," *Nature communications*, vol. 9, no. 1, p. 4812, 2018.

[12] Z. Holmes *et al.*, "Connecting ansatz expressibility to gradient magnitudes and barren plateaus," *PRX quantum*, vol. 3, no. 1, p. 010313, 2022.

[13] S. Wang *et al.*, "Noise-induced barren plateaus in variational quantum algorithms," *Nature communications*, vol. 12, no. 1, p. 6961, 2021.

[14] A. Arrasmith, Z. Holmes, M. Cerezo, and P. J. Coles, "Equivalence of quantum barren plateaus to cost concentration and narrow gorges," *Quantum Science and Technology*, vol. 7, no. 4, p. 045015, 2022.

[15] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 9, pp. 5149–5169, 2021.

[16] J. Lee, J. Cho, and S. Kim, "Q-maml: Quantum model-agnostic meta-learning for variational quantum algorithms," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, pp. 18137–18144, 2025.

[17] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, "Learning to learn by gradient descent by gradient descent," *Advances in neural information processing systems*, vol. 29, 2016.

[18] S. Y.-C. Chen *et al.*, "Quantum convolutional neural networks for high energy physics data analysis," *Physical Review Research*, vol. 4, no. 1, p. 013231, 2022.

[19] K.-C. Chen, S. Y.-C. Chen, C.-Y. Liu, and K. K. Leung, "Toward large-scale distributed quantum long short-term memory with modular quantum computers," in *2025 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 337–342, IEEE, 2025.

[20] Y.-C. Hsu *et al.*, "Quantum kernel-based long short-term memory," in *2025 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pp. 1–5, IEEE, 2025.

[21] Y.-C. Hsu *et al.*, "Quantum kernel-based long short-term memory for climate time-series forecasting," in *2025 International Conference on Quantum Communications, Networking, and Computing (QCNC)*, pp. 421–426, IEEE, 2025.

[22] Y.-C. Hsu *et al.*, "Federated quantum kernel-based long short-term memory for human activity recognition," *arXiv preprint arXiv:2508.06078*, 2025.

[23] S. Y.-C. Chen, "Learning to program variational quantum circuits with fast weights," in *2024 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9, IEEE, 2024.

[24] S. Y.-C. Chen *et al.*, "Learning to program quantum measurements for machine learning," *arXiv preprint arXiv:2505.13525*, 2025.

[25] C.-Y. Liu *et al.*, "Programming variational quantum circuits with quantum-train agent," in *2025 International Conference on Quantum Communications, Networking, and Computing (QCNC)*, pp. 544–548, IEEE, 2025.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] L. R. Medsker, L. Jain, *et al.*, "Recurrent neural networks," *Design and applications*, vol. 5, no. 64-67, p. 2, 2001.

[28] Siami-Namini *et al.*, "Forecasting economics and financial time series: Arima vs. lstm," *arXiv preprint arXiv:1803.06386*, 2018.

[29] A. Srivastava *et al.*, "Weather prediction using lstm neural networks," in *2022 IEEE 7th International conference for Convergence in Technology (I2CT)*, pp. 1–4, IEEE, 2022.

[30] Z. Shi *et al.*, "Lstm-based flight trajectory prediction," in *2018 International joint conference on neural networks (IJCNN)*, pp. 1–8, IEEE, 2018.

[31] L. Yao and Y. Guan, "An improved lstm structure for natural language processing," in *2018 IEEE international conference of safety produce informatization (IICSPI)*, pp. 565–569, IEEE, 2018.

[32] S. Wang and J. Jiang, "Learning natural language inference with lstm," *arXiv preprint arXiv:1512.08849*, 2015.

[33] M. Lippi *et al.*, "Natural language statistical features of lstm-generated texts," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3326–3337, 2019.

[34] E. Azari and S. Vrudhula, "An energy-efficient reconfigurable lstm accelerator for natural language processing," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 4450–4459, IEEE, 2019.

[35] Z. C. Lipton *et al.*, "Learning to diagnose with lstm recurrent neural networks," *arXiv preprint arXiv:1511.03677*, 2015.

[36] D. C. Edara, L. P. Vanukuri, V. Sistla, and V. K. K. Kolli, "Sentiment analysis and text categorization of cancer medical records with lstm," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 5, pp. 5309–5325, 2023.

[37] P. Choppara and B. Lokesh, "Leveraging quantum lstm for high-accuracy prediction of viral mutations," *IEEE Access*, 2025.

[38] A. S. Younger, S. Hochreiter, and P. R. Conwell, "Meta-learning with backpropagation," in *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, vol. 3, IEEE, 2001.

[39] S. Y.-C. Chen, C.-H. H. Yang, J. Qi, P.-Y. Chen, X. Ma, and H.-S. Goan, "Variational quantum circuits for deep reinforcement learning," *IEEE access*, vol. 8, pp. 141007–141024, 2020.

[40] S. Khairy *et al.*, "Learning to optimize variational quantum circuits to solve combinatorial problems," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 2367–2375, 2020.

[41] Y.-C. Hsu *et al.*, "Quantum adaptive excitation network with variational quantum circuits for channel attention," *arXiv preprint arXiv:2507.11217*, 2025.

[42] K.-C. Chen *et al.*, "Validating large-scale quantum machine learning: Efficient simulation of quantum support vector machines using tensor networks," *Machine Learning: Science and Technology*, 2024.

[43] S. Thanasilp *et al.*, "Exponential concentration in quantum kernel methods," *Nature communications*, vol. 15, no. 1, p. 5200, 2024.

[44] Z.-L. Tsai *et al.*, "Learning the hierarchy of steering measurement settings of qubit-pair states with kernel-based quantum models," *New Journal of Physics*, vol. 27, no. 9, p. 094502, 2025.

[45] J. Schmidhuber, "Learning to control fast-weight memories: An alternative to dynamic recurrent networks," *Neural Computation*, vol. 4, no. 1, pp. 131–139, 1992.

[46] A. Antipov, *Efficient Realization of Quantum Primitives for Shor's Algorithm Using PennyLane Library v1*. ZappyLab, Inc., 2022. Available online: https://pennylane.ai/ (Accessed: Jul. 19, 2025).

[47] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[48] P. Roy, "Scipy tools plenary on scipy," in *Proc. Python in Science Conf. (SciPy)*, NumFOCUS and Insight Software Consortium (ITK), 2023.

[49] G. Verdon, M. Broughton, J. R. McClean, K. J. Sung, R. Babbush, Z. Jiang, H. Neven, and M. Mohseni, "Learning to learn with quantum neural networks via classical neural networks," *arXiv preprint arXiv:1907.05415*, 2019.

[50] R. Pascanu *et al.*, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on Machine Learning* (S. Dasgupta and D. McAllester, eds.), vol. 28 of *Proceedings of Machine Learning Research*, (Atlanta, Georgia, USA), pp. 1310–1318, PMLR, 17–19 Jun 2013.

[51] I. Goodfellow *et al.*, *Deep Learning*. MIT Press, 2016.

[52] D. Sherrington and S. Kirkpatrick, "Solvable model of a spin-glass," *Phys. Rev. Lett.*, vol. 35, pp. 1792–1796, Dec 1975.

[53] I. D. Kivlichan *et al.*, "Quantum simulation of electronic structure with linear depth and connectivity," *Phys. Rev. Lett.*, vol. 120, p. 110501, Mar 2018.

[54] R.-Y. Chang, Y.-C. Lin, P.-C. Hsu, T.-W. Huang, and E.-J. Kuo, "Accelerating parameter initialization in quantum chemical simulations via lstm-fc-vqe," *IEEE Access*, pp. 1–1, 2025.