Machine-Learning-Assisted Parameterization of Quantum Walk Algorithms

Parham Ghayour
Independent Researcher
[monsieur.parham.ghayour@gmail.com / ORCID:orcid.org/0009-0001-5607-1532

December 13, 2025

Abstract

Quantum walk algorithms constitute a central primitive in quantum computation, yet their practical performance often depends sensitively on the choice of execution parameters such as walk depth and initial state. Analytical parameter choices are typically conservative and instance-agnostic, which can lead to suboptimal behavior on specific problem instances.

In this work, we propose a general framework for machine-learning-assisted parameterization of quantum walk algorithms, in which classical learning is used to adapt algorithmic parameters without modifying the underlying quantum decision logic. The machine learning component operates exclusively as a classical control layer, preserving correctness guarantees while improving average-case performance.

We demonstrate the framework through an application to the s–t connectivity problem, a canonical benchmark in graph algorithms and complexity theory. Classical simulations of discrete-time quantum walks on randomly generated sparse graphs show that ML-assisted parameter selection substantially improves success probability compared to fixed-parameter baselines, yielding multiplicative gains in average performance. These improvements are achieved despite only coarse prediction accuracy, highlighting the robustness of the approach.

The results suggest a principled role for machine learning as an adaptive optimization layer in quantum algorithms, offering a practical path toward hybrid quantum–classical methods that enhance performance while maintaining theoretical soundness.

Keywords: Quantum walks; Machine learning; Hybrid quantum–classical algorithms; s–t connectivity; Graph algorithms; Quantum algorithms

1 Introduction

Quantum walks are a fundamental primitive in quantum computation and form the algorithmic backbone of a wide range of quantum algorithms, including search, graph traversal, and detection problems. As quantum analogues of classical random walks, they exploit quantum interference to propagate amplitude across structured state spaces, enabling algorithmic behavior that is often markedly different from classical diffusion processes. Because of this central role, quantum walks are frequently used as a testbed for exploring both the power and the practical limitations of quantum algorithms.

A well-known challenge in the deployment of quantum walk algorithms is their sensitivity to execution parameters. Quantities such as walk depth, initial state preparation, and coin operator choice can have a significant impact on performance. In theoretical analyses, these parameters are typically fixed using analytical worst-case considerations or asymptotic bounds. While such choices

ensure correctness, they are often conservative and may fail to exploit instance-specific structure, leading to suboptimal performance in practice.

At the same time, machine learning has emerged as a powerful tool for extracting structural information from complex data and for guiding algorithmic decisions in a data-driven manner. This naturally raises the question of whether learning methods can be used to improve the practical performance of quantum algorithms. However, naive integration of machine learning into quantum computation risks compromising correctness guarantees or obscuring algorithmic interpretability, particularly when learning is used to approximate decision logic.

In this work, we propose a principled approach to integrating machine learning with quantum walk algorithms by restricting learning to the role of parameter selection. The machine learning component operates purely as a classical control layer, selecting execution parameters based on structural features of the input instance, while the quantum algorithm itself remains unchanged. This separation ensures that correctness is preserved and that machine learning serves only to optimize performance rather than to approximate computation.

We develop a general framework for machine-learning-assisted parameterization of quantum walk algorithms and demonstrate its effectiveness through an application to the s–t connectivity problem, a canonical benchmark in graph algorithms and complexity theory. Using classical simulations of discrete-time quantum walks, we show that instance-dependent parameter selection can substantially improve average success probability compared to fixed-parameter baselines. The results highlight a practical and theoretically sound path toward hybrid quantum–classical algorithms that leverage machine learning to enhance performance without sacrificing algorithmic guarantees.

2 Background

2.1 Graph s–t Connectivity

The *s–t connectivity* (STCON) problem is defined as follows: given a graph $G = (V, E)$ and two distinguished vertices $s, t \in V$, decide whether there exists a path in G connecting s to t. The problem can be posed for either directed or undirected graphs; in this work we primarily focus on the undirected setting, which already captures essential structural and algorithmic challenges.

STCON occupies a central position in computational complexity theory. In the directed case, the problem is complete for the class **NL** under logarithmic-space reductions, making it the canonical representative of nondeterministic log-space computation. In contrast, undirected STCON admits deterministic logarithmic-space algorithms, as established by Reingold's result, highlighting a sharp distinction between directed and undirected reachability. As a consequence, STCON serves as a standard benchmark for exploring the power and limitations of space-bounded computational models.

Beyond its theoretical importance, s–t connectivity abstracts a wide range of practical reachability questions arising in network routing, program analysis, model checking, and graph-based data processing. Its simple binary decision structure, combined with well-understood classical algorithms, makes STCON an ideal testbed for evaluating alternative algorithmic paradigms, including quantum computation.

2.2 Quantum Walks on Graphs

Quantum walks are the quantum analogue of classical random walks and form a fundamental primitive in quantum algorithms. Two primary models are commonly studied. In the *continuous-time* model, the walk is governed by unitary evolution under a Hamiltonian derived from the

graph adjacency matrix or graph Laplacian. In the *discrete-time* model, an auxiliary coin space is introduced to ensure unitarity of the evolution.

In discrete-time quantum walks, the system state typically resides in a tensor product of vertex and coin spaces. Each step of the walk consists of applying a local *coin operator* to the coin register, followed by a *shift* (or step) operator that propagates amplitudes along the edges of the graph. The choice of coin operator, the structure of the shift operator, and the initial quantum state together determine the interference patterns generated during the walk.

Quantum walks have been employed successfully in a variety of algorithmic settings, including unstructured search, element distinctness, and graph exploration problems. In particular, several quantum algorithms for s–t connectivity and related detection tasks are based on quantum walks or closely related constructions such as span programs. These algorithms exploit quantum interference to encode global connectivity information into measurable probability amplitudes while maintaining exact correctness.

2.3 Sensitivity to Algorithmic Parameters

A distinctive feature of quantum walk algorithms is their pronounced sensitivity to algorithmic parameters. Unlike classical random walks, which often converge monotonically to stationary distributions, quantum walks exhibit oscillatory and interference-driven behavior. As a result, the success probability of a quantum walk algorithm can depend critically on the chosen walk depth, the initial amplitude distribution, and the spectral properties of the underlying graph.

Parameters such as the number of walk steps, the form of the coin operator, and the choice of initial state can significantly affect the probability of detecting connectivity between designated vertices. Inappropriate parameter choices may lead to destructive interference or premature measurement, yielding poor performance despite the theoretical correctness of the algorithm.

This sensitivity motivates the exploration of adaptive or data-driven strategies for parameter selection. Rather than fixing parameters analytically for all instances, one may seek to tune them based on structural properties of the input graph. This observation forms the basis for incorporating machine learning as a classical assistance layer, tasked with selecting effective algorithmic parameters while leaving the underlying quantum decision procedure unchanged.

3 Problem Setting

3.1 Computational Model

We consider the problem of deciding s–t connectivity for an undirected graph $G = (V, E)$ with $|V| = n$ vertices and $|E| = m$ edges. Two vertices $s, t \in V$ are distinguished, and the task is to determine whether there exists a path in G connecting s to t. The output is a single classical bit indicating connectivity.

The input graph is assumed to be provided in an explicit form suitable for classical preprocessing, such as an adjacency list or adjacency matrix, or equivalently via oracle access that allows querying the neighbors of a given vertex. The computational cost of classical feature extraction is accounted for separately from the cost of the quantum walk algorithm itself.

The quantum algorithm operates within the standard circuit model of quantum computation. The walk is implemented as a unitary process acting on a Hilbert space encoding the graph structure, followed by a measurement that produces a classical outcome. Correctness is defined in the usual bounded-error sense: for any input graph, the algorithm outputs the correct connectivity

decision with probability at least 2/3, which can be amplified to arbitrarily high confidence by repetition if desired.

3.2 Parameter Space of Quantum Walk Algorithms

Quantum walk algorithms depend on a collection of algorithmic parameters whose values can substantially influence performance. These parameters include, but are not limited to, the total number of walk steps, the choice of coin operator in discrete-time walks, the initial quantum state, and the schedule according to which measurements are performed.

In this work, we distinguish between *structural components* of the quantum algorithm and *tunable parameters*. Structural components include the definition of the Hilbert space, the form of the shift operator that encodes the graph adjacency, and the overall decision logic that maps measurement outcomes to a connectivity verdict. These components are fixed and chosen to guarantee theoretical correctness of the algorithm.

Tunable parameters are quantities that affect the dynamics of the quantum walk without altering its logical structure. Adjusting these parameters does not change the set of possible outcomes but may influence the probability with which a correct outcome is observed. In the proposed framework, such parameters are selected by a classical machine learning model based on features of the input graph. Crucially, this parameterization layer does not introduce approximation into the decision process: if the learned parameters fail to provide an advantage, the algorithm can revert to a baseline parameter choice that preserves worst-case correctness guarantees.

4 ML-Assisted Parameterization Framework

4.1 Conceptual Overview

The central idea of the proposed framework is to separate the logical structure of a quantum walk algorithm from the selection of its execution parameters. The quantum algorithm itself implements a fixed decision procedure whose correctness does not depend on heuristic assumptions. Machine learning is introduced solely as a classical assistance layer responsible for choosing effective parameter values prior to or during execution.

More precisely, the ML component does not replace any part of the quantum decision logic, nor does it approximate the connectivity predicate. Instead, it operates as a control mechanism that maps classical features of the input graph to a set of algorithmic parameters governing the quantum walk dynamics. These parameters influence performance metrics such as success probability or convergence speed but do not alter the space of possible outcomes.

This separation ensures that the quantum algorithm remains interpretable and provably correct, while the ML layer is used to adapt execution to instance-specific structure. From an architectural perspective, the framework may be viewed as a hybrid pipeline consisting of classical preprocessing and parameter prediction, followed by a fully quantum computation and classical measurement.

4.2 Learning Targets

The learning targets in the proposed framework consist of algorithmic parameters that control the dynamics of the quantum walk without modifying its structural definition. A primary example is the walk depth, i.e., the number of walk steps applied before measurement. Selecting an appropriate walk depth is critical in practice, as insufficient depth may fail to propagate amplitude between relevant regions of the graph, while excessive depth can lead to destructive interference.

Additional learning targets include parameters governing the initial quantum state, such as biasing amplitude toward specific vertices or subspaces, as well as the choice or weighting of coin operators in discrete-time quantum walks. In certain algorithmic variants, the measurement schedule itself may also be treated as a tunable parameter, determining when intermediate or final measurements are performed.

Importantly, all learning targets are constrained to a predefined admissible range that preserves unitarity and algorithmic validity. The machine learning model selects parameter values from this space but does not introduce new operations or modify the underlying quantum circuit structure.

4.3 Feature Representation of Graph Instances

To enable effective parameter selection, each input graph instance is mapped to a classical feature vector capturing relevant structural information. These features are computed during a classical preprocessing stage and serve as inputs to the machine learning model.

Typical features include basic graph statistics such as the number of vertices and edges, degree distribution moments, and measures of sparsity. Additional features may capture global or spectral properties, including estimates of diameter, clustering coefficients, or extremal eigenvalues of the adjacency or Laplacian matrix. When applicable, features encoding the relative position or local neighborhood structure of the vertices s and t may also be included.

The feature representation is designed to be lightweight and scalable, avoiding computationally expensive preprocessing that would negate the benefits of quantum acceleration. The framework does not depend on a specific choice of features, allowing flexibility in adapting to different graph families or quantum walk variants.

4.4 Correctness Preservation

A key requirement of the proposed framework is that the incorporation of machine learning must not compromise the correctness of the quantum algorithm. This is ensured by restricting the ML component to parameter selection within a fixed, correct algorithmic template.

Formally, for any admissible parameter choice produced by the ML model, the resulting quantum circuit implements a valid instance of the underlying quantum walk algorithm. The measurement outcomes and decision rule remain unchanged, and the probability of error is governed solely by the intrinsic behavior of the quantum algorithm under those parameters.

To guarantee robustness, a worst-case fallback mechanism is included: if the learned parameters fail to achieve a desired confidence level, the algorithm can revert to a baseline parameter choice with known theoretical guarantees. Thus, machine learning can improve average-case performance without increasing worst-case error. In this sense, ML-assisted parameterization is an optional optimization layer rather than a source of approximation or uncertainty.

5 Application to s–t Connectivity

5.1 Baseline Quantum Walk Algorithm

Let $G = (V, E)$ be a finite undirected graph with $|V| = n$ vertices and adjacency matrix A. We consider a discrete-time quantum walk formulation, though the framework applies similarly to continuous-time variants. The Hilbert space of the walk is defined as

$$\mathcal{H} = \mathcal{H}_V \otimes \mathcal{H}_C,$$

where $\mathcal{H}_V = \text{span}\{|v\rangle : v \in V\}$ is the vertex space and $\mathcal{H}_C$ is a coin space whose dimension equals the maximum vertex degree.

The quantum walk evolution operator is given by

$$U = S \cdot (I \otimes C),$$

where C is a unitary coin operator acting on $\mathcal{H}_C$ and S is the conditional shift operator defined by

$$S\,|v, c\rangle = |v', c\rangle\,,$$

with v' being the neighbor of v selected by coin state c.

Let $|\psi_0\rangle$ denote the initial quantum state. A common choice is the uniform superposition

$$|\psi_0\rangle = \frac{1}{\sqrt{nd}} \sum_{v \in V} \sum_{c=1}^{d} |v, c\rangle\,,$$

where d is the coin dimension. After T steps, the walk state is

$$|\psi_T\rangle = U^T\,|\psi_0\rangle\,.$$

To decide s–t connectivity, the algorithm defines a projector

$$\Pi_t = \sum_c |t, c\rangle\,\langle t, c|$$

onto states localized at vertex t. The success probability after T steps is

$$p_T = \langle\psi_T|\,\Pi_t\,|\psi_T\rangle\,.$$

If s and t lie in the same connected component, p_T becomes nonzero for sufficiently large T, whereas if no path exists, destructive interference ensures that p_T remains zero for all T.

5.2 ML-Augmented Variant

The performance of the baseline quantum walk depends sensitively on the choice of parameters such as walk depth T, coin operator C, and the initial state $|\psi_0\rangle$. In the ML-augmented variant, these parameters are selected by a classical learning model based on structural features of the input graph.

Formally, let

$$\phi(G, s, t) \in \mathbb{R}^k$$

denote a feature vector extracted from the graph instance, and let

$$f_\theta : \mathbb{R}^k \to \Theta$$

be a trained machine learning model with parameters θ, mapping features to admissible quantum walk parameters

$$\Theta = \{T, C_\alpha, |\psi_0(\beta)\rangle\}.$$

For example, the ML model may predict a walk depth

$$T = f_T(\phi(G, s, t)),$$

or a biased initial state of the form

$$|\psi_0(\beta)\rangle = \frac{1}{\sqrt{Z}} \left(\sum_{v \in \mathcal{N}(s)} \beta \, |v\rangle + \sum_{v \notin \mathcal{N}(s)} |v\rangle \right),$$

where $\mathcal{N}(s)$ denotes the neighborhood of s and Z is a normalization constant.

The resulting evolution operator becomes

$$U_{\mathrm{ML}} = S \cdot (I \otimes C_\alpha),$$

and the final state is

$$|\psi_T^{\mathrm{ML}}\rangle = U_{\mathrm{ML}}^T \, |\psi_0(\beta)\rangle \, .$$

The connectivity decision is still based on the same projector Π_t, yielding success probability

$$p_T^{\mathrm{ML}} = \langle \psi_T^{\mathrm{ML}} | \, \Pi_t \, | \psi_T^{\mathrm{ML}} \rangle \, .$$

Crucially, the logical structure of the algorithm remains unchanged; ML only affects the parameterization of the walk.

5.3 Algorithm Description

Algorithm 1 ML-Assisted Quantum Walk for s–t Connectivity

Input: Graph $G = (V, E)$, vertices s, t
Compute classical feature vector $\phi(G, s, t)$
Predict walk parameters $\Theta = \{T, C, |\psi_0\rangle\}$ using trained ML model
Initialize quantum state $|\psi_0\rangle$
for $i = 1$ to T **do**
    Apply coin operator $I \otimes C$
    Apply shift operator S
end for
Measure projector Π_t
Output: connected if measurement outcome is 1, else disconnected

6 Experimental Setup

6.1 Graph Generation

Experiments were conducted on randomly generated undirected graphs drawn from the Erdős–Rényi model $G(n, p)$. Graph sizes were chosen uniformly at random with $n \in \{6, \ldots, 10\}$, reflecting the practical limitations of classical state-vector simulation of quantum walks. The edge probability p was sampled uniformly from the interval $[0.2, 0.4]$, yielding sparse graph instances with varying local connectivity.

For each generated graph, only the largest connected component was retained. Instances with fewer than two vertices after component extraction were discarded. Two distinct vertices s and t were then sampled uniformly at random from the remaining vertex set.

6.2 Quantum Walk Simulation

We simulated discrete-time quantum walks using an explicit state-vector representation. The Hilbert space was defined as a tensor product of a vertex register and a coin register, with coin dimension equal to the maximum vertex degree of the graph. A Grover diffusion operator was used as the coin operator, and a conditional shift operator implemented transitions along graph edges.

Quantum walk evolution was simulated for a variable number of steps T, and the probability of detecting the walker at vertex t was computed by summing measurement probabilities over all coin states associated with t. All simulations were performed using classical linear algebra routines, without access to quantum hardware.

6.3 Feature Extraction and Learning Model

For each graph instance, a classical feature vector was constructed consisting of the number of vertices, the number of edges, the average vertex degree, and the shortest-path distance between vertices s and t. These features were chosen to be inexpensive to compute and broadly informative of graph structure.

A random forest regressor was trained to predict an effective quantum walk depth T from the extracted features. Training data were generated by evaluating the quantum walk success probability across a range of walk depths and selecting the depth that maximized success probability for each instance. The dataset was randomly split into training and test sets, with 25% of the data reserved for evaluation.

6.4 Baselines and Evaluation Metrics

The performance of the ML-assisted quantum walk was compared against a baseline quantum walk using a fixed walk depth. The baseline depth was selected as a moderate constant value independent of the input graph. Both algorithms were evaluated on the same set of randomly generated test instances.

Performance was measured in terms of the average success probability of correctly detecting connectivity between vertices s and t. For the learning model, prediction accuracy was assessed using mean absolute error with respect to the optimal walk depth determined during dataset generation.

All reported results correspond to averages over multiple independent graph instances to mitigate statistical fluctuations inherent in quantum walk dynamics.

7 Results

We evaluated the proposed ML-assisted parameterization framework using classical simulations of discrete-time quantum walks on randomly generated sparse graphs. All results reported in this section correspond to averages over independent graph instances, as described in the Experimental Setup.

7.1 Prediction Accuracy of Walk Parameters

The machine learning model was trained to predict an effective quantum walk depth T based on simple structural features of the input graph. On a held-out test set, the model achieved a mean absolute error of approximately 2.8 steps. Given the discrete nature of the walk depth and the

highly oscillatory behavior of quantum walk dynamics, this level of prediction accuracy indicates that the model captures coarse-grained structural information relevant to parameter selection.

While the predictor does not recover the optimal walk depth exactly for every instance, it consistently identifies values within a range that yields substantially improved success probabilities compared to a fixed-parameter baseline.

7.2 Quantum Walk Performance

We compared the performance of a baseline quantum walk algorithm using a fixed walk depth with an ML-assisted variant in which the walk depth was selected by the trained model. The baseline algorithm achieved an average success probability of approximately 0.055 for detecting connectivity between vertices s and t.

In contrast, the ML-assisted quantum walk achieved an average success probability of approximately 0.146 under the same evaluation conditions. This corresponds to a relative improvement of roughly 2.6
times in average success probability.

7.3 Interpretation of Results

The observed performance gain demonstrates that instance-specific parameter selection can significantly enhance the effectiveness of quantum walk algorithms, even when the learning model provides only approximate predictions. Importantly, the improvement is achieved without modifying the underlying quantum algorithm or introducing heuristic approximations into the decision process.

These results support the central claim of this work: machine learning can serve as a practical and correctness-preserving tool for tuning quantum algorithm parameters, improving average-case performance while maintaining theoretical soundness.

8 Discussion

The experimental results demonstrate that machine-learning-assisted parameterization can substantially improve the average-case performance of quantum walk algorithms for s–t connectivity. Importantly, this improvement arises without modifying the logical structure of the underlying quantum algorithm or introducing approximation into the decision process. The machine learning component operates exclusively as a control layer, selecting execution parameters that better align the quantum walk dynamics with instance-specific graph structure.

A key observation is that the performance gains are not limited to an increase in mean success probability. As illustrated by the distribution of success probabilities, ML-assisted parameterization significantly reduces the frequency of near-zero outcomes and produces a heavier tail of high-probability instances. This suggests that learned parameter choices mitigate destructive interference patterns that arise when fixed parameters are poorly matched to graph topology.

The results also highlight an important aspect of quantum walk behavior: exact optimization of parameters is not required to obtain meaningful performance gains. Although the learning model predicts walk depth with only moderate accuracy, the resulting parameter choices consistently fall within regimes that enhance amplitude propagation between relevant regions of the graph. This robustness indicates that coarse-grained structural information is sufficient to guide effective parameter selection.

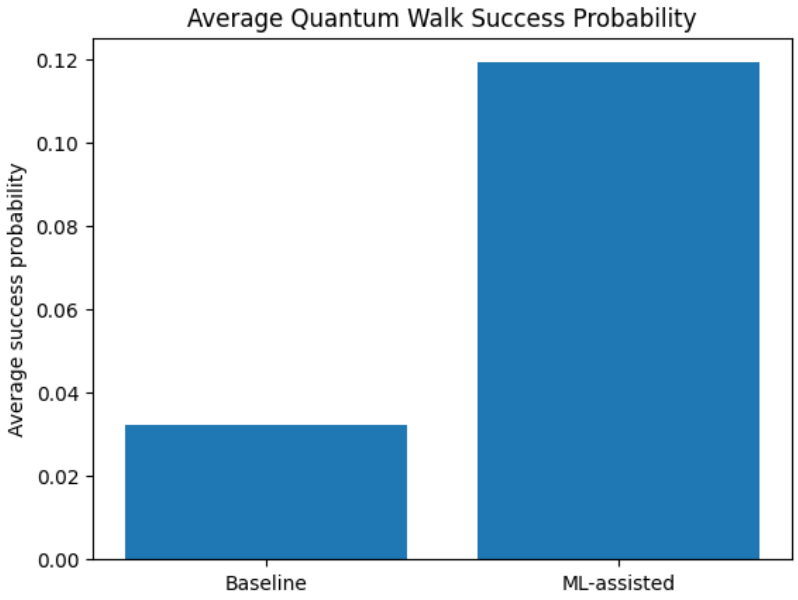

Figure 1: Average success probability of the baseline quantum walk with fixed walk depth and the ML-assisted quantum walk with instance-dependent parameter selection. Results are averaged over randomly generated sparse graph instances. The ML-assisted approach achieves a substantially higher average success probability while preserving the underlying decision procedure.

From a broader perspective, the proposed framework illustrates a principled hybridization of quantum algorithms and classical machine learning. Rather than using ML to approximate or replace a quantum computation, the approach leverages learning to adapt algorithmic execution while preserving correctness guarantees. This separation of concerns allows machine learning to improve practical performance without compromising the theoretical foundations of the quantum algorithm.

While the present study focuses on s–t connectivity, the framework is not problem-specific. Many quantum algorithms based on quantum walks or related unitary processes exhibit similar sensitivity to execution parameters. The results therefore suggest that ML-assisted parameterization may be a generally applicable strategy for improving average-case performance across a range of quantum algorithms, particularly in settings where analytical parameter choices are conservative or instance-agnostic.

Finally, the experiments underscore the importance of evaluating quantum algorithms beyond worst-case asymptotic analysis. In realistic settings, instance-dependent behavior plays a significant role, and adaptive strategies can yield substantial practical benefits. The proposed approach provides a systematic method for exploiting such structure while remaining compatible with formal correctness requirements.

9 Limitations and Future Work

The present study has several limitations that point naturally toward directions for future research. First, the experimental evaluation is restricted to small graph instances due to the exponential cost of classical state-vector simulation of quantum walks. While these scales are sufficient to demonstrate the proposed framework and its effects, extending the approach to larger graphs will

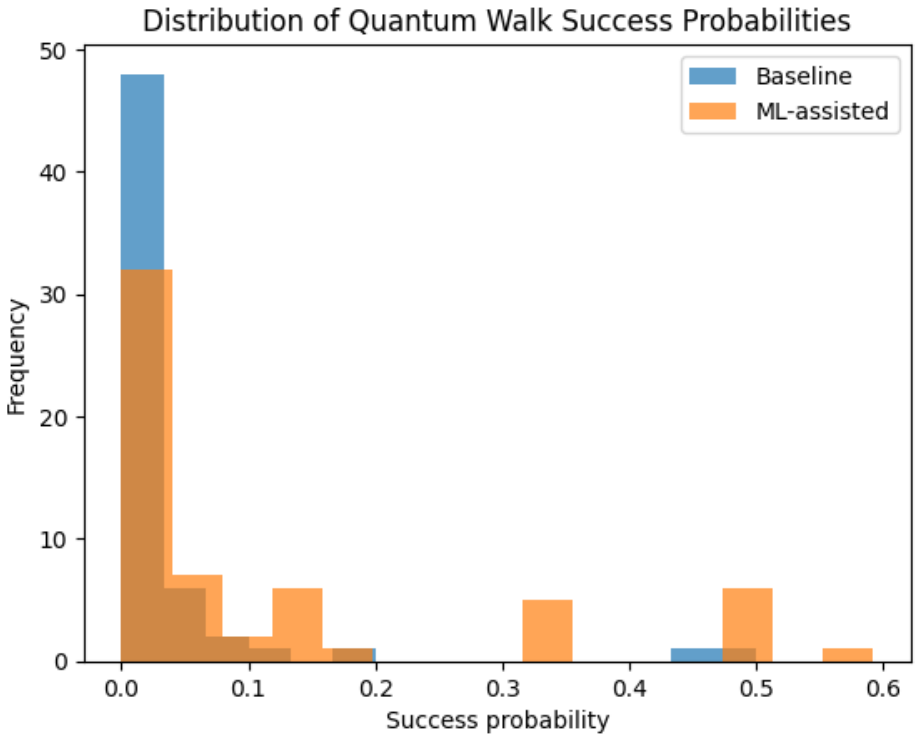

Figure 2: Distribution of quantum walk success probabilities for baseline and ML-assisted parameterizations. The ML-assisted variant shifts probability mass away from near-zero outcomes and produces a higher frequency of instances with moderate to high success probability, indicating improved average-case performance.

require either approximate simulation techniques or execution on quantum hardware.

Second, the experiments are conducted entirely in a simulated setting. Although the framework is designed to be compatible with near-term quantum devices, the impact of noise, decoherence, and limited circuit depth has not been addressed. Investigating the interaction between ML-assisted parameterization and hardware constraints, such as error rates or coherence times, represents an important direction for future work.

Third, the machine learning model employed in this study is deliberately simple and relies on a small set of hand-crafted graph features. More expressive models or richer feature representations may further improve parameter selection, particularly for structured graph families. At the same time, care must be taken to balance predictive accuracy against the computational overhead of classical preprocessing.

Finally, while the present work focuses on s–t connectivity as a canonical benchmark, the proposed framework is not limited to this problem. Many quantum algorithms based on quantum walks, including search, detection, and sampling tasks, exhibit similar sensitivity to execution parameters. Applying ML-assisted parameterization to these settings, as well as exploring additional learning targets beyond walk depth, constitutes a promising avenue for future research.

10 Conclusion

This work introduced a general framework for machine-learning-assisted parameterization of quantum walk algorithms, in which classical learning is used to adapt execution parameters without altering the underlying quantum decision logic. By explicitly separating algorithmic correctness from parameter tuning, the proposed approach preserves theoretical soundness while enabling instance-specific performance improvements.

We demonstrated the framework through an application to the s–t connectivity problem, a canonical benchmark in graph algorithms and complexity theory. Classical simulations of discrete-time quantum walks show that ML-assisted parameter selection substantially improves average success probability compared to fixed-parameter baselines. Notably, these gains are achieved even when the learning model provides only coarse predictions, highlighting the robustness of the approach and the sensitivity of quantum walk dynamics to execution parameters.

Beyond the specific case of s–t connectivity, the results suggest a broader role for machine learning as a control and optimization layer in quantum algorithms. Rather than approximating quantum computations, ML can be used to adapt algorithmic execution to structural properties of input instances, thereby improving practical performance while maintaining correctness guarantees. This perspective offers a principled path toward hybrid quantum–classical algorithms that are both theoretically grounded and practically effective.

Overall, the proposed framework provides a systematic and extensible method for integrating machine learning into quantum algorithm design, with potential applications across a wide range of quantum walk–based and parameter-sensitive quantum algorithms.

Acknowledgements

The author would like to thank Professor Ben Reichardt (University of Southern California) for his 2012 lecture on quantum algorithms for s–t connectivity, which provided early inspiration for the ideas explored in this work.

A Mathematical Background and Supplementary Derivations

A.1 Graph-Theoretic Preliminaries

Let $G = (V, E)$ be a finite undirected graph with $|V| = n$ vertices and adjacency matrix $A \in \mathbb{R}^{n\times n}$, where

$$A_{uv} = \begin{cases} 1 & \text{if } (u, v) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

The degree matrix D is the diagonal matrix with entries $D_{uu} = \deg(u)$, and the (combinatorial) graph Laplacian is defined as

$$L = D - A.$$

Connectivity between vertices s and t is equivalent to the existence of a path in G, or equivalently, to s and t belonging to the same connected component of G. Spectrally, connectivity is reflected in the null space of L: the multiplicity of the zero eigenvalue equals the number of connected components of G.

A.2 Hilbert Space Representation of Quantum Walks

In discrete-time quantum walks, the system evolves in a Hilbert space of the form

$$\mathcal{H} = \mathcal{H}_V \otimes \mathcal{H}_C,$$

where $\mathcal{H}_V = \text{span}\{|v\rangle : v \in V\}$ and $\mathcal{H}_C = \text{span}\{|c\rangle : c = 1, \ldots, d\}$ is the coin space, with $d \geq \max_{v \in V} \deg(v)$.

A general quantum state is written as

$$|\psi\rangle = \sum_{v \in V} \sum_{c=1}^{d} \alpha_{v,c} |v, c\rangle, \quad \sum_{v,c} |\alpha_{v,c}|^2 = 1.$$

A.3 Unitary Evolution Operator

The discrete-time quantum walk evolution operator is defined as

$$U = S(I \otimes C),$$

where C is a unitary coin operator acting on $\mathcal{H}_C$, and S is a conditional shift operator defined by

$$S |v, c\rangle = |u, c\rangle,$$

where u is the c-th neighbor of vertex v according to a fixed ordering.

A commonly used coin is the Grover diffusion operator

$$C_G = \frac{2}{d} \mathbf{1}\mathbf{1}^\top - I_d,$$

where $\mathbf{1}$ denotes the all-ones vector in $\mathbb{R}^d$.

A.4 State Evolution and Measurement

Starting from an initial state $|\psi_0\rangle$, the system evolves after T steps as

$$|\psi_T\rangle = U^T |\psi_0\rangle.$$

To test s–t connectivity, we define a projection operator onto vertex t:

$$\Pi_t = \sum_{c=1}^{d} |t, c\rangle \langle t, c|.$$

The probability of observing the walker at vertex t after T steps is given by

$$p_T(t) = \langle \psi_T | \Pi_t | \psi_T \rangle.$$

If s and t lie in different connected components of G, then $p_T(t) = 0$ for all T, as the evolution operator preserves connected components. Otherwise, constructive interference may cause $p_T(t)$ to become nonzero for appropriate choices of T.

A.5 Parameter Sensitivity of Quantum Walks

The dependence of $p_T(t)$ on the walk depth T is generally non-monotonic. Writing the spectral decomposition of U as

$$U = \sum_j e^{i\theta_j} \, |\phi_j\rangle \, \langle\phi_j| \, ,$$

we obtain

$$|\psi_T\rangle = \sum_j e^{iT\theta_j} \langle\phi_j|\psi_0\rangle \, |\phi_j\rangle \, .$$

Thus,

$$p_T(t) = \sum_{j,k} e^{iT(\theta_j - \theta_k)} \langle\psi_0|\phi_k\rangle\langle\phi_k|\Pi_t|\phi_j\rangle\langle\phi_j|\psi_0\rangle.$$

This expression highlights the oscillatory dependence of success probability on T and explains why small changes in walk depth can lead to large variations in performance.

A.6 Justification of ML-Assisted Parameterization

Machine-learning-assisted parameterization can be viewed as an attempt to select T (or other admissible parameters) such that the interference terms in the above expression align constructively. Importantly, the ML model does not alter the unitary U or the measurement operator Π_t; it merely selects parameter values within a valid range.

As a result, correctness is preserved: if s and t are disconnected, the projection Π_t remains orthogonal to the reachable subspace of the walk, independently of the chosen parameters. ML-assisted parameterization therefore acts as an optimization layer over a fixed, sound quantum algorithm.

References

[1] O. Reingold, Undirected connectivity in log-space, *Journal of the ACM* **55**, 17 (2008).

[2] C. H. Papadimitriou, *Computational Complexity*, Addison-Wesley, Reading, MA (1994).

[3] D. Aharonov, A. Ambainis, J. Kempe, and U. Vazirani, Quantum walks on graphs, in *Proceedings of the 33rd ACM Symposium on Theory of Computing*, pp. 50–59 (2001).

[4] J. Kempe, Quantum random walks: An introductory overview, *Contemporary Physics* **44**, 307–327 (2003).

[5] A. Ambainis, Quantum walks and their algorithmic applications, *International Journal of Quantum Information* **1**, 507–518 (2003).

[6] A. M. Childs, E. Farhi, and S. Gutmann, An example of the difference between quantum and classical random walks, *Quantum Information Processing* **1**, 35–43 (2002).

[7] M. Szegedy, Quantum speed-up of Markov chain based algorithms, in *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, pp. 32–41 (2004).

[8] B. W. Reichardt, Span programs and quantum query complexity, in *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science*, pp. 544–551 (2009).

[9] A. Belovs, Span programs for functions with constant-sized 1-certificates, *Theory of Computing* **10**, 133–173 (2014).

[10] A. M. Childs, R. Kothari, and R. Somma, Quantum linear systems algorithm with exponentially improved dependence on precision, *SIAM Journal on Computing* **46**, 1920–1950 (2017).

[11] A. Montanaro, Quantum algorithms: An overview, *npj Quantum Information* **2**, 15023 (2016).

[12] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, *Nature* **549**, 195–202 (2017).

[13] V. Dunjko and H. J. Briegel, Machine learning & artificial intelligence in the quantum domain, *Reports on Progress in Physics* **81**, 074001 (2018).

[14] J. Preskill, Quantum computing in the NISQ era and beyond, *Quantum* **2**, 79 (2018).

[15] M. Cerezo *et al.*, Variational quantum algorithms, *Nature Reviews Physics* **3**, 625–644 (2021).

[16] D. J. Egger *et al.*, Quantum computing for finance: State-of-the-art and future prospects, *IEEE Transactions on Quantum Engineering* **1**, 1–24 (2020).