Generative QAOA and Its Application to Portfolio Optimization

Luan Costa and Gabriel Coutinho

***Abstract*— We present the Generative Quantum Approximate Optimization Algorithm (GQAOA), a method for applying classical generative models to optimize QAOA circuit parameters. Our work is inspired by the Generative Quantum Eigensolver (GQE) proposed by Nakaji et al. [1], which trains a classical generative model to produce a sequence of quantum gates from a predefined set to generate quantum states that minimize the energy of a Hamiltonian. Our method adapts this framework to optimize the QAOA circuit parameters, given an Ising Hamiltonian as input that represents a real-world portfolio optimization problem.**

***Keywords*— Quantum Approximate Optimization Algorithm (QAOA), Generative Models, Portfolio Optimization, Quadratic Unconstrained Binary Optimization (QUBO), Ising Hamiltonian.**

I. Introduction

The promise of quantum computing lies in its potential to revolutionize fields in which classical computation struggles, particularly in solving complex optimization problems. In the financial sector, a key challenge is portfolio optimization, which seeks to find the best balance between risk and return. Although fully fault-tolerant quantum computers are still on the horizon, the current era of near-term quantum devices necessitates the use of hybrid quantum-classical algorithms that leverage the strengths of both computational paradigms. A leading example of this approach is the Quantum Approximate Optimization Algorithm (QAOA), a hybrid algorithm designed to find approximate solutions to combinatorial optimization problems, such as the Quadratic Unconstrained Binary Optimization (QUBO) problem that underpins portfolio optimization.

Traditionally, the optimization of QAOA circuit parameters has relied on classical, gradient-free methods, such as the Nelder-Mead simplex algorithm or COBYLA (Constrained Optimization By Linear Approximations). These methods iteratively explore the parameter landscape without directly using the gradient of an objective function. A key limitation of this approach is its inefficiency in dealing with the sequential dependencies between the parameters across different QAOA layers, which can lead to slow convergence and a high risk of becoming trapped in local minima, particularly as the circuit depth increases.

Currently, a variety of Large Language Models (LLMs) are available that yield impressive results in text completion. These models employ a concept known as the attention mechanism [2] to capture context, thereby generating more coherent sequences of tokens within the realm of language. However, this architecture can be adapted to address other types of problems involving the generation of sequential data.

Inspired by this new paradigm, Nakaji et al. [1] proposed the Generative Quantum Eigensolver (GQE), which applies a classical generative model to produce sequences of quantum gates for the Variational Quantum Eigensolver (VQE) algorithm [5]. The VQE is another hybrid algorithm aimed at finding the ground-state energy of a Hamiltonian.

In this study, we introduce a Generative Quantum Approximate Optimization Algorithm (GQAOA), which adapts a generative framework to optimize the QAOA parameters for the quantum portfolio optimization problem. Our approach involves training a classical generative model to learn and predict the optimal parameter sequences. By incorporating an adaptive training loop that directly utilizes past measurement energies from the quantum circuit, we guided the generative model to produce parameters that resulted in lower energy states. This method addresses the limitations of traditional optimization techniques by effectively leveraging the sequential nature of the parameters and considering the model output related to the measured energy of the circuit, which enables the updates through backpropagation.

We applied the method to a portfolio optimization problem, a well-known financial problem that seeks to find the optimal allocation of assets to maximize returns while minimizing risk. We modeled this as a Quadratic Unconstrained Binary Optimization (QUBO) problem, which is a common approach for converting this type of problem into a form that can be solved by quantum optimization algorithms such as the QAOA [7]. This allowed us to demonstrate our generative approach by identifying a portfolio that effectively balances risk and return. Our results show that the GQAOA framework successfully learns the optimal parameters for a basic example, providing proof of concept for the use of generative models to enhance quantum optimization algorithms in real-world financial applications.

II. Portfolio Optimization Problem

The portfolio optimization problem is a central challenge in financial investment, and aims to allocate capital among various assets to achieve the most favorable balance between risk and expected return. This is a fundamental concept in modern portfolio theory (MPT), which was first introduced by Markowitz [3]. MPT established that a portfolio's risk and return are not simply the sum of its individual assets but also depend on their co-movements or correlations.

A core principle of Markowitz's work is the trade-off between expected returns and risk. An investor can generally

Luan Costa, DCC, UFMG, BH-MG, e-mail: luan.costa@dcc.ufmg.br; Gabriel Coutinho, DCC, UFMG, BH-MG, e-mail: gabriel@dcc.ufmg.br.

achieve a higher expected return only by taking on greater risk. Therefore, the goal of portfolio optimization is to find portfolios that either maximize the expected return for a given level of risk or minimize the risk for a given expected return. This set of optimal portfolios forms an efficient frontier, a curve representing all portfolios that yield the best possible return-to-risk ratio. Any portfolio that lies below the efficient frontier is considered suboptimal because it either provides a lower return for the same amount of risk or the same return for a higher amount of risk.

In this study, we simplified this problem to better convert it to run on a quantum device. We will address the problem of identifying the best assets to invest in, given a list of possibilities and a predefined parameter that captures the investor's appetite for risk. To mathematically define this problem, we introduce matrix Q as given in Equation 1.

$$Q = -R + q \cdot Cov. \tag{1}$$

In this equation, Q is a matrix that encapsulates the balance between expected returns and risk. The term R is a diagonal matrix containing the expected returns for each asset, and it drives the portfolio toward assets with higher potential profits. Cov is the covariance matrix of the assets, where the entry (i, j) represents the covariance between assets i and j. This term accounts for the risk and diversification benefits of asset combination. The scalar q is a risk aversion parameter. A higher value of q signifies that an investor is more sensitive to risk, thus increasing the weight of the covariance term in the optimization.

Using matrix Q, we can define the following cost function:

$$C(x) = x^T Q x. \tag{2}$$

The cost function under consideration in this study is a quadratic form that we aim to minimize. The vector x denotes the investment decisions, is a binary vector such that $x_i = 1$ if asset i is included in the portfolio, and $x_i = 0$ if it is not. The objective is to identify the binary vector x that minimizes $C(x)$, thereby selecting the optimal set of assets for the portfolio based on the specified risk-return criteria. Consequently, this constitutes a Quadratic Unconstrained Binary Optimization (QUBO) problem.

Finally, we impose a constraint to control the portfolio size. The number of selected assets must be equal to a predefined value B. This is expressed as the sum of binary variables:

$$\sum_i x_i = B. \tag{3}$$

III. Quantum Approximate Optimization Algorithm (QAOA)

The Quantum Approximate Optimization Algorithm (QAOA) [4] is a hybrid quantum-classical algorithm designed to solve combinatorial optimization problems, particularly those that can be formulated as Quadratic Unconstrained Binary Optimization (QUBO) problems. This makes it a highly valuable tool in finance, where such problems often arise in tasks such as portfolio optimization and risk management. QAOA operates by leveraging the principles of quantum mechanics and using a quantum system to search for possible solutions more efficiently than classical methods.

The core of the QAOA lies in its ability to encode the problem into a Hamiltonian, a mathematical object that represents the energy of a quantum system. The goal is to find the ground state of this Hamiltonian, which corresponds to the optimal solution to the problem. To achieve this, the QAOA alternates between two key quantum operations: a phase operator that encodes the objective function and a mixing operator that helps explore different possible configurations of the system.

The phase operator is expressed mathematically as $U(\gamma) = e^{-i\gamma H_C}$, where H_C is the problem Hamiltonian representing the cost function, and γ is a parameter that controls the phase rotation. This operation shifts the phase of the quantum state based on the value of the cost function, effectively guiding the system toward lower-energy configurations, which correspond to better solutions.

Next, the mixing operator $U(\beta)$ is applied to explore the solution space. In [4], it is given by $U(H_B, \beta) = e^{-i\beta H_B}$, where H_B is the mixing Hamiltonian, usually defined as $B = \sum_i X_i$, with X_i representing the Pauli-X matrix acting on the $i - th$ qubit. The role of this operator is to induce transitions between different possible states, thereby allowing the algorithm to sample a wide range of configurations. The parameter β determines the extent to which the system "mixes," helping to avoid local minima and improving the chances of finding the global optimum.

The QAOA algorithm alternates between applying the phase and mixing operators, with each layer characterized by a pair of parameters (γ_i, β_i). The quantum state after p layers (depth of the circuit) is described by the following equation:

$$|\psi(\gamma, \beta)\rangle = U_B(\beta_p)U_C(\gamma_p)\dots U_B(\beta_1)U_C(\gamma_1)\,|\psi_0\rangle\,. \tag{4}$$

Here, ψ_0 is the initial state of the system, which is typically chosen to be an equal superposition of all possible solutions, allowing the algorithm to start with a broad search across the solution space.

To evaluate the quality of the solution, the expectation value of the cost function is computed as

$$C(\gamma, \beta) = \langle\psi(\gamma, \beta)|\, H_C\, |\psi(\gamma, \beta)\rangle\,. \tag{5}$$

This expectation value is minimized using a classical optimization algorithm that adjusts the parameters (γ, β) iteratively. The goal is to determine the set of parameters that minimize the cost, which corresponds to finding the best possible solution to the optimization problem.

When dealing with QUBO problems, the cost Hamiltonian can be constructed from the QUBO formulation itself. Once this Ising Hamiltonian is obtained, it can serve as the cost Hamiltonian in the QAOA algorithm, which is applied during the phase operation to guide the quantum state toward an optimal solution.

A. Cost Hamiltonian for Portfolio Optimization

To establish a correspondence between the cost function 2 and an equivalent Ising model, the binary variables x_i

are transformed into spin operators through a straightforward mapping, $x_i \to (I - Z_i)/2$. This transformation enables the reformulation of the QUBO problem as an Ising Hamiltonian 6, where the binary optimization task is equivalent to determining the ground state of the spin system [7]:

$$H_C = \sum_i (-\sum_j Q_{ij}) Z_i + \sum_{i>j} Q_{ij} Z_i Z_j. \tag{6}$$

B. The Mixture Hamiltonian and Initial State Preparation

To address the constraint specified in Eq. 3, we employ the Dicke state [6] as the initial state of the QAOA circuit, Eq. 7. This state is a superposition of all computational basis states with exactly B excited qubits out of a total of n qubits:

$$|\psi_0\rangle = |D_B^n\rangle = \frac{1}{\sqrt{\binom{n}{B}}} \sum_{\substack{i_1,\ldots,i_n=0,1 \\ i_1+\cdots+i_n=B}} |i_1, \ldots, i_n\rangle . \tag{7}$$

The Dicke state is an eigenstate of the operator $(X_iX_j + Y_iY_j)$ for all pairs of qubits i and j. As a result, this operator preserves the total number of ones. This conservation property allows us to construct a mixing unitary that does not move the system out of the constrained subspace. Therefore, we define our mixing unitary using rotations based on this operator, as expressed in Eq. 8 [7].

$$R_{i,j}(\beta) = e^{i\beta(X_iX_j + Y_iY_j)}. \tag{8}$$

The full mixing unitary, $U_B(\beta)$, is then constructed by applying these rotations to a selected set of connected qubit pairs, E. For this study, we choose E to form a ring graph, where $E = \{(1,2),(2,3),\ldots,(n,n+1)\}$, and qubit $n+1$ is identified with qubit 1 [7].

$$U_B(\beta) = \prod_{(i,j)\in E} R_{i,j}(\beta). \tag{9}$$

IV. Generative QAOA

In this study, we propose a novel approach for QAOA parameter optimization by training the GPT-2 model using the Hugging Face library to predict QAOA parameter sequences. The model learns to associate specific parameter sequences with the corresponding expected values of the cost Hamiltonian. By incorporating the measured energy from the quantum circuit into the training loop, we can guide the generative model to produce parameter sequences that lead to lower-energy states.

A. Quantum circuits generation

To enable the GPT model to generate the parameters, we first discretized the possible values of γ_i and β_i into s discrete possibilities:

$$\gamma_i \in \left\{0, \frac{1}{s-1}\cdot 2\pi, \ldots, 2\pi, \right\},$$
$$\beta_i \in \left\{0, \frac{1}{s-1}\cdot 2\pi, \ldots, 2\pi, \right\}.$$

For each layer k of the QAOA circuit, where $k \geq 2$, the GPT model receives a sequence of previous parameter indices $(0, j_1, j_2, \ldots, j_{2k-3}, j_{2k-2})$ and predicts the indices for the current layer, j_{2k-1} and j_{2k}, which correspond to γ_k and β_k. For the first layer ($k = 1$), the model receives only index 0 as a starting point.

The process works as follows, in each layer k, GPT generates two sets of logits values

$$\mathcal{W}'^{(k)} = \{w'_1, w'_2, \ldots, w'_s\},$$
$$\mathcal{W}''^{(k)} = \{w''_1, w''_2, \ldots, w''_s\},$$

for all possible values of γ_k and β_k. A sampling function then generates the indices of layer k based on a softmax probability distribution, such that

$$Pr(j_i) \propto \begin{cases} \exp(-w'_i/T) & \text{if } i \leq s, \\ \exp(-w''_{s+i}/T) & \text{if } i > s, \end{cases}$$

where T is the temperature parameter. Let $w'^{(k)}(\vec{j})$ and $w''^{(k)}(\vec{j})$ be the generated logits for the indices chosen by the sampling function. The sample function applied in each layer generates the following probability of a general list of index $\vec{j}$ being generated:

$$Pr(\vec{j}) = \frac{1}{Z}\exp(-w_{sum}(\vec{j})/T), \tag{10}$$

where $w_{sum}(\vec{j}) = \sum_k (w'^{(k)}(\vec{j}) + w''^{(k)}(\vec{j}))$ [1].

We can also provide the GPT model with a fixed list of indices, $\vec{j}$, to obtain the logit values $w'^{(k)}(\vec{j})$ and $w''^{(k)}(\vec{j})$ generated for each respective layer, which is important for training.

B. Training

Suppose we have two index lists, $\vec{j_1}$ and $\vec{j_2}$, and we obtain their corresponding logit sums from the GPT model, $w_{sum}(\vec{j_1})$ and $w_{sum}(\vec{j_2})$. We can then execute the QAOA circuit with the parameters $(\gamma(\vec{j}), \beta(\vec{j}))$ corresponding to these indices and measure the expected value of the cost Hamiltonian in Eq. 5, $C(\vec{j}) = C(\gamma(\vec{j}), \beta(\vec{j}))$.

In this case, we defined the loss function as

$$L = (C(\vec{j_1}) - w_{sum}(\vec{j_1}))^2 + (C(\vec{j_2}) - w_{sum}(\vec{j_2}))^2.$$

Suppose that the indices in $\vec{j_1}$ produce a state with lower energy than $\vec{j_2}$ but the GPT model has not yet learned this, that is, $w_{sum}(\vec{j_2}) < w_{sum}(\vec{j_1})$. By updating the model's internal parameters via backpropagation, a successful update results in an increased probability of generating indices $\vec{j_1}$, described by Eq. 10, while the probability of generating $\vec{j_2}$ would decrease.

For each training epoch, we propose a specific method for selecting the index lists. We select three lists:

1) Positive-temperature sample: We use the sampling function with a defined temperature T to generate a list of indices, which explores the low-energy predictions of the model.
2) Negative-temperature sample: We use the sampling function with a negative temperature $-T$ to generate a

second list. This explores the model's high-energy predictions, enabling the model to be adjusted by penalizing inaccurate predictions.

3) Best-so-far list: The third list is the set of indices that has produced the lowest energy measured on the quantum circuit up to the current epoch. This "best-so-far" list forces the model to converge towards the empirically best-performing parameters found so far, accelerating the optimization process.

In this methodology, the number of executions of the QAOA circuit is equivalent to twice the number of epochs. This is because the quantum circuit is executed twice for each epoch to measure the energy produced by the positive and negative temperature samples. In the case of the "best-so-far" list, the previously computed energy value was utilized.

V. Results

To test the proposed approach, we used the returns and covariance matrix in tables I and II, the same used in [7].

TABLE I
RETURN VECTOR, ROUNDED TO 2 DECIMAL PLACES, FOR 5 ASSETS CHOSEN FROM THE GERMAN DAX 30

LIN.DE	BAYN.DE	VNA.DE	MTX.DE	MUV2.DE
0.27	−0.12	0.21	0.22	0.11

TABLE II
COVARIANCE MATRIX, ROUNDED TO 2 DECIMAL PLACES, FOR 5 ASSETS CHOSEN FROM THE GERMAN DAX 30: LIN.DE, BAYN.DE, VNA.DE, MTX.DE AND MUV2.DE.

0.21	0.03	0.01	0.03	0.03
0.03	0.09	0.02	0.05	0.04
0.01	0.02	0.05	0.02	0.02
0.03	0.05	0.02	0.14	0.06
0.03	0.04	0.02	0.06	0.07

We trained GPT to predict QAOA parameters depth 10, with 30 epochs and the temperature parameter $T = 10$. With the result parameters, we measured the state generated by the circuit 1000 times, and the histogram with the results is shown in Figure 1; the most frequent state $|10100\rangle$ represents the portfolio with the assets LIN.DE and MTX.DE, which is the optimal state, as can be verified using brute force in this simple case, measuring the cost function in Eq. 2 for all 10 portfolios that satisfies the constraint in Eq. 3.

We trained the GPT model for 30 epochs, implying that the quantum circuit was executed 60 times. We performed the optimization using a traditional method, COBYLA optimization, which also generated the optimal state as the most frequent result, as shown in the histogram of figure 2, but in that case, the quantum circuit was executed 191 times. Therefore, in that basic case, we obtained a 68% reduction in the number of calls in the quantum circuit.

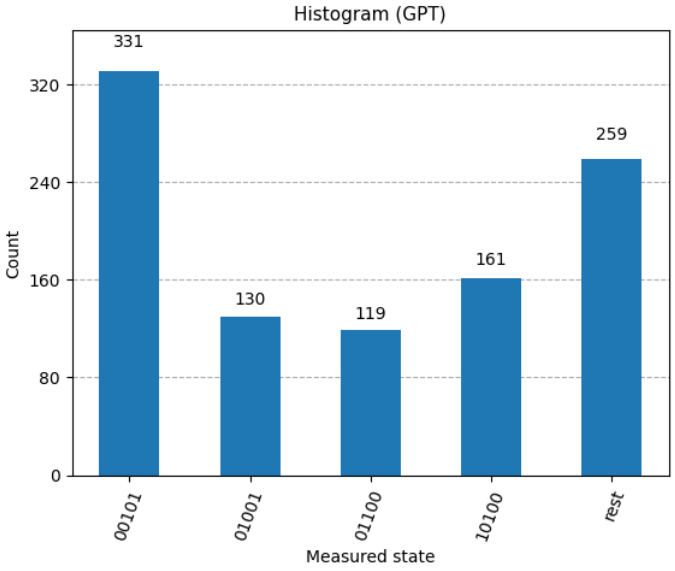

Fig. 1. The histogram illustrates the states produced by the Quantum Approximate Optimization Algorithm (QAOA), with parameters optimized via the GPT model. The circuit was executed 1000 times to generate the measured states in the histograms. The states are presented in reverse order due to the notation conventions of Qiskit.

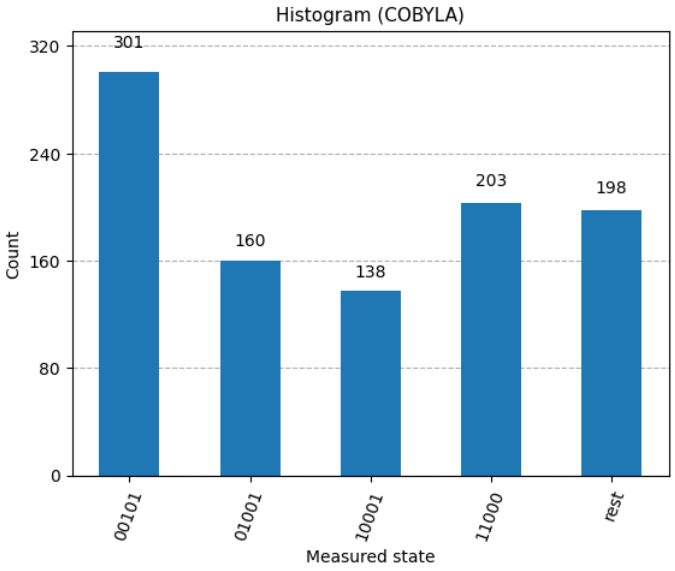

Fig. 2. The histogram illustrates the states produced by the Quantum Approximate Optimization Algorithm (QAOA), with parameters optimized via the COBYLA optimizer. The circuit was executed 1000 times to generate the measured states in the histograms. The states are presented in reverse order due to the notation conventions of Qiskit.

VI. Conclusion

In this study, we illustrated the effectiveness of the Generative Quantum Approximate Optimization Algorithm (GQAOA) in a basic scenario as a strategy for optimizing the QAOA parameters. By utilizing the GPT-2 model to generate the parameter sequences of QAOA for a portfolio optimization problem involving five assets, with the constraint that the optimal portfolio comprises two assets, our

method reduced the number of quantum circuit executions necessary to determine the optimal portfolio compared with the conventional COBYLA optimizer. In the problem under consideration, GQAOA was successfully optimized with 60 quantum circuit calls, marking a 68% improvement over the 191 calls required by COBYLA. This highlights the potential of generative models in optimizing quantum circuit parameters by considering the interdependence of these parameters.

ACKNOWLEDGMENTS

Luan Costa acknowledges finantial support from FAPEMIG. Gabriel Coutinho acknowledges support from FAPEMIG, CNPq and Banco Inter.

REFERENCES

[1] K. Nakaji, L. B. Kristensen, J. A. Campos-Gonzalez-Angulo, M. G. Vakili, H. Huang, M. Bagherimehrab, C. Gorgulla, F. Wong, A. McCaskey, J.-S. Kim et al., *The generative quantum eigensolver (GQE) and its application for ground state search*. arXiv:2401.09253, 2024.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. *Attention is all you need*. Advances in Neural Information Processing Systems 30, 2017.

[3] Harry Markowitz, "Portfolio Selection," *The Journal of Finance*, v. 7, pp. 77–91, 1952.

[4] Farhi, E., Goldstone, J., Gutmann, S., *A Quantum Approximate Optimization Algorithm*. arXiv:1411.4028, 2014

[5] Peruzzo, A. and McClean, J. and Shadbolt, P. "A variational eigenvalue solver on a photonic quantum processor," *Nature Communications*, vol. 5, pp. 4213, 2014.

[6] Dicke, R.H., "Coherence in spontaneous radiation processes," *Phys. Rev.*, v. 93, pp. 99–110, 1954.

[7] Brandhofer, S., Braun, D., D., Dehn, V., Hellstern, G., Hüls, M., Ji, Y., Polian, I., Bhatia, A.S., & Wellens, T., "Benchmarking the performance of portfolio optimization with QAOA," *Quantum Information Processing*, v. 22, pp. 52–55, 2022.