Q3SAT-GPT: A Generative Model for Discovering Quantum Circuits for the 3-SAT Problem

Pratim Ugale*, Ilya Tyagin*, Karunya Shirali†‡, Kien X. Nguyen*, Ilya Safro*§
*Department of Computer and Information Sciences, University of Delaware, Newark, DE, USA
†Department of Physics, Virginia Tech, Blacksburg, VA, USA
‡Virginia Tech Center for Quantum Information Science and Engineering, Blacksburg, VA, USA
§Department of Physics and Astronomy, University of Delaware, Newark, DE, USA
{pratim, tyagin, kxnguyen, isafro}@udel.edu, karunyashirali@vt.edu

***Abstract*—This work introduces Q3SAT-GPT, a generative model for discovering quantum circuits for the Max-E3-SAT problem. Our method learns from high-performing QAOA-style ansätze to directly generate candidate circuits. To create high-quality supervision, we also introduce Mosaic Adaptive QAOA (MosaicADAPT-QAOA), an adaptive strategy for constructing low-depth QAOA circuits by selecting subsets of mixer operators in each step, rather than inserting operators sequentially. The resulting circuits serve as training data for the generative model, allowing it to learn effective circuit design patterns while eliminating the need for costly variational optimization at inference time. Experiments show that our framework attains strong solution quality with shallow circuits and scales significantly better than both our adaptive construction procedure and conventional variational baselines. Our results establish generative modeling as a high-performance route toward the scalable discovery of quantum optimization circuits, demonstrating that these models can effectively internalize circuit logic while providing a foundation for future, instance-aware inductive biases. Reproducibility: The source code is available at https://github.com/pratimugale/Q3SAT-GPT.**

***Index Terms*—Q3SAT-GPT, MosaicADAPT-QAOA, QAOA, Max3SAT, QAOA-GPT, Quantum Optimization, Generative AI**

I. Introduction

Quantum computing faces a central scientific bottleneck: useful fault-tolerant machines will not deliver impact through hardware alone, but through the discovery of new algorithms that exploit problem structure (without which possible complexity improvements are severely limited [1]–[3]) while respecting such resource constraints as logical qubits, circuit depth, and error-corrected compilation [4]–[6]. Quantum algorithms for combinatorial optimization are not an exception. While application domains for it are broad spanning finance [7], network science [8], and computational biology [9] to mention just a few, at present, most quantum algorithms for optimization are hand-designed, or adapted from generic templates. This is especially limiting for combinatorial problems, where naive encodings often destroy structure, inflate resource counts, and erase any plausible advantage. The critical challenge, therefore, is not merely to run known quantum routines on better hardware, but to create AI systems that can discover reduced-complexity, fault-tolerant quantum algorithms tailored to the problems of interest, together with compilation and verification tools that make those algorithms executable and trustworthy [10], [11].

The goal of this work is to design a generative AI framework for one of the most broadly applicable NP-hard optimization problems: Max-E3-SAT [12] (the problem of maximizing the number of satisfied clauses in CNF-SAT instances with exactly three literals in each clause). Max-E3-SAT (also referred to as 3-SAT) is tightly connected to constraint satisfaction and has motivated decades of research in both exact and approximate classical optimization, resulting in a large ecosystem [13], [14] of highly engineered solvers and benchmarks. This makes it an especially meaningful testbed for generative quantum algorithm design: any proposed quantum-circuit synthesis method must be evaluated against a problem class for which classical baselines are strong, and extensively studied. Another aspect of interest is that, similarly to MaxCut (a traditional target for quantum combinatorial optimization due to its direct connection to the Ising model), Max-E3-SAT guarantees simple but strong approximation ratios which are by themselves not bad, e.g., Håstad showed that, unless $\mathrm{P} = \mathrm{NP}$, no polynomial algorithm can guarantee an approximation ratio better than $7/8+\varepsilon$ for any fixed $\varepsilon > 0$ [12]. Thus, Max-E3-SAT provides an excellent setting for studying whether generative AI can discover problem-aware quantum circuit structures that go beyond templates and well known approximations.

The Quantum Approximate Optimization Algorithm (QAOA) [15] has emerged as a leading variational framework for combinatorial optimization, with the potential to provide advantages for certain problem classes or instance families. Its near-term relevance stems from its hybrid quantum-classical structure: a parameterized quantum circuit prepares candidate solution states, while a classical optimizer updates the variational parameters. This architecture makes QAOA particularly attractive for both noisy intermediate-scale quantum and fault-tolerant regimes. However, original QAOA and most of its subsequent versions [16] have a rigid circuit structure, which directly impacts expressibility and inability to cope with barren plateaus and can substantially degrade solution quality before circuits reach depths sufficient to exploit problem structure.

Several variants of QAOA have been proposed to improve different aspects of the original algorithm. Parameter transfer-

ability and initialization strategies aim to reduce the classical optimization overhead by exploiting concentration and transferability of good angles across related instances [17]–[19]. Warm-start QAOA incorporates solutions of classical relaxations into the initial quantum state [20]. Multi-angle QAOA increases expressivity by assigning independent parameters to different terms in the cost and mixer Hamiltonians [21]. Sparsification of the phase operator reduces the number of gates in QAOA [22]. Recursive QAOA combines variational optimization with iterative variable elimination [23]. In this sense, QAOA is not merely a quantum algorithm, but a hybrid optimization paradigm whose success depends on carefully balancing quantum expressivity with classical control.

The most relevant to this work are adaptive approaches that change the circuit structure [24]–[26]. In particular, ADAPT-QAOA constructs the ansatz problem-dependently by selecting operators from a pool according to gradient information. More recent extension, TETRIS-ADAPT further seeks to reduce circuit depth and optimization cost by selecting multiple operators per layer [27].

In this work, we develop a GPT-based generative approach for discovering quantum algorithms for Max-E3-SAT. Rather than treating quantum circuit design as a purely hand-crafted or repeatedly optimized variational task, our framework learns circuit generation patterns from high-quality adaptive circuits and then produces candidate circuits directly by autoregressive generation. To construct the training corpus, we introduce MosaicADAPT-QAOA, a modification of the TETRIS-ADAPT strategy that selects compatible sets of mixer operators at each layer *by optimizing their collective contribution*, rather than selecting the set of mixer operators to add at each layer by greedily inserting operators one at a time as in TETRIS-ADAPT. This produces shallow, problem-adapted circuits that serve as supervision for the generative model. The resulting Q3SAT-GPT framework is therefore designed to amortize the cost of slow adaptive circuit construction and variational parameter optimization: the expensive MosaicADAPT-QAOA procedure is used offline to generate training data, while inference requires only a single forward generative pass to propose a quantum circuit for a new Max-E3-SAT instance.

Another challenge in QAOA is the optimization of the parameters in the ansatz. The optimization involves measuring the expectation energy of the circuit, and updating the parameters to optimize the expectation. While adaptive methods lead to shallower circuits by constructing problem-tailored ansätze, repeated energy gradient measurements are a significant overhead to the computational cost of the algorithm. Even after selecting the best operator, the parameters of the quantum circuit must again be optimized, along with the parameters associated with the newly added operator. Similar to our previous work QAOA-GPT [28], in Q3SAT-GPT we also generate fully optimized quantum circuits end-to-end.

In summary, our contributions are:

1) **MosaicADAPT-QAOA:** The dense tiling method that extends ADAPT and TETRIS methods. The tiling is designed to optimize the sum of gradients of disjoint operators. For example, we observe that MosaicADAPT-QAOA requires a median of 4 fewer layers to reach 99.9% approximation ratio on 10 variable problems.
2) **Q3SAT-GPT:** We introduce a GPT-based generative framework for synthesizing quantum circuits for Max-E3-SAT instances. Using high-quality circuits produced by MosaicADAPT-QAOA as supervision, the model learns structural and parametric patterns that characterize effective adaptive ansätze for this problem class. Once trained, Q3SAT-GPT generates candidate circuits in a single autoregressive inference pass. This positions Q3SAT-GPT as a scalable AI generative approach for quantum circuit discovery, providing strong generated circuits for Max-E3-SAT while opening a path toward instance-aware generative design of quantum optimization algorithms.

II. Background

A. The Max-E3-SAT Problem

The Max-E3-SAT problem [29] is an NP-hard combinatorial optimization problem [30]. The goal is to maximise the number of satisfied clauses in a formula in its Conjunctive Normal Form (CNF), where each clause comprises of a disjunction (logical OR) of exactly three literals.

Let $\mathcal{F} = C_1 \wedge \cdots \wedge C_m$ be a CNF formula containing m clauses over n Boolean variables $x_i \in \{0, 1\}$. A literal l is defined as either a variable x_i or its negation $\neg x_i$. A clause C_r is represented as:

$$C_r = (l_{r,1} \vee l_{r,2} \vee l_{r,3}),$$

and is satisfied if at least one of its constituent literals evaluates to 1 (True). The goal of a Max-E3-SAT solver is to find a variable assignment that maximizes:

$$\sum_{r=1}^{m} \mathbb{1}[C_r \text{ is satisfied}],$$

where $\mathbb{1}[\cdot]$ is the indicator function. A simple randomized algorithm that independently assigns truth values to variables with a probability of $1/2$ satisfies an expected 87.5% of the clauses, as each clause in a 3-CNF formula has a $7/8$ probability of containing at least one true literal. This random baseline is useful when interpreting approximation ratios, especially for satisfiable instances where the optimum equals the total number of clauses.

Uniform Random Formulas. The Uniform Random 3-SAT model defines a specific distribution of CNF formulas [31]. In this approach, a formula is constructed by independently generating m clauses over $2n$ literals, while discarding clauses with redundant literals or contradictory variable pairs.

Balanced Formulas. The generation protocol for balanced Max-E3-SAT instances strictly enforces variable regularity, which means that the total frequency of each variable across the formula varies by no more than a single occurrence (± 1) [32]. This rule also applies to literal polarity, providing an

almost perfectly equal split between positive variables and their negations, while minimizing structural overlap between clauses.

Critical Phase Transition Ratio. The difficulty and satisfiability of Max-E3-SAT instances is governed by its phase transition ratio, defined as the ratio of number of clauses in formula to the number of distinct variables $\alpha = m/n$. The probability of an instance being satisfiable undergoes a sharp decline as the clause count m for a given number of variables n increases toward the critical phase transition ratio [31], [33]. For Uniform Random 3-SAT instances, the value $\alpha \approx 4.26$ marks the critical phase transition between satisfiable and unsatisfiable regimes [31], while for Balanced instances, this phase transition ratio α is found to be at 3.6 [32].

B. The Quantum Approximate Optimization Algorithm

The Quantum Approximate Optimization Algorithm (QAOA) [34] aims to find a bitstring solution $x \in \{0,1\}^n$ that optimizes a classical function. The classical objective function is first encoded into a cost Hamiltonian H_C, typically formulated as an Ising model consisting of Pauli-Z operators. The algorithm initializes n qubits in a uniform superposition $|+\rangle^{\otimes n}$. The parameterized QAOA ansatz is constructed by applying p alternating layers of the cost Hamiltonian H_C and a mixing Hamiltonian H_M, parameterized by $2p$ angles $\boldsymbol{\gamma} = \{\gamma_i\}_{i=1}^p$ and $\boldsymbol{\beta} = \{\beta_i\}_{i=1}^p$:

$$|\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle = \prod_{k=1}^{p} \left(e^{-i\beta_k H_M} e^{-i\gamma_k H_C}\right) |s\rangle$$

To optimize the state, a classical optimizer iteratively updates $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ to minimize the expectation value of the cost Hamiltonian:

$$E(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \langle \psi(\boldsymbol{\gamma}, \boldsymbol{\beta}) | H_C | \psi(\boldsymbol{\gamma}, \boldsymbol{\beta}) \rangle$$

Achieving high quality solutions with QAOA typically require a high number of layers, which is challenging to execute on current quantum hardware. Previous studies have proposed parameter-transfer and initialization strategies to reduce the optimization time by exploiting concentration and transferability of good angles across related instances [19], [35], [36].

C. Adaptive Methods for Ansätze Creation

Methods such as ADAPT-VQE [37] and ADAPT-QAOA [24] address the limitations of current devices and the locality constraints of standard QAOA [38] by adaptively and iteratively building ansätze tailored to the specific problem. By selecting only the most impactful operators, these algorithms achieve accelerated convergence. ADAPT-QAOA replaces the standard fixed mixer in QAOA with a pool of operators $\mathcal{P}$. At each layer of QAOA, the algorithm selects one operator $\hat{A}_j$ from this pool based on an adaptive selection process. The selected operator is the one whose addition to the circuit is estimated to produce the largest immediate improvement in the QAOA objective.

In contrast to the standard QAOA mixer which acts on all qubits, the operator pool in ADAPT-QAOA consists of single-qubit and two-qubit Pauli operators. The operator selection process for ADAPT-QAOA [24] was originally inspired by ADAPT-VQE [37], which selects a single operator per layer. TETRIS-ADAPT-VQE [27] is a recent development that allows the selection of multiple disjoint operators in a single step and has been shown to result in significantly shallower circuits compared to ADAPT-VQE. This is because multiple operators can be added to the ansatz at the same step, allowing the expressivity to grow while preventing any qubit from sitting idle at a particular layer.

D. QAOA-GPT

While algorithmic variations like ADAPT-QAOA dynamically tailor the ansatz to the problem instance, the iterative evaluation of gradients at each layer, along with the classical optimization of the variational parameters, incurs a significant classical computational overhead. To address this bottleneck, recent literature has explored data-driven, machine learning frameworks. QAOA-GPT [28] is a generative pre-trained transformer model designed to synthesize quantum circuits in a single inference step.

QAOA-GPT was trained on a large dataset of graph instances paired with their corresponding optimized circuits generated via standard ADAPT-QAOA. The model bypasses the iterative gradient calculations and parameter optimization entirely. While such generative approaches offer massive speedups at inference time, their performance is inherently bounded by the quality of the training data. Therefore, advancing the underlying adaptive algorithms that generate this training data, remains a critical necessity for the scalable success of AI-driven quantum circuit generation.

III. Methodology

A. Overview of the Pipeline

The proposed methodology has two stages. First, MosaicADAPT-QAOA constructs high-quality adaptive QAOA circuits for Max-E3-SAT by selecting compatible sets of mixer operators at each layer. These circuits are then used as supervision for Q3SAT-GPT, which conditions on an input 3-CNF formula and its Literal-Clause Graph (LCG) embeddings to generate a tokenized MosaicADAPT-QAOA-style circuit in a single autoregressive pass.

B. Problem Formulation and Hamiltonian Encoding

To evaluate the algorithm's performance on unapproximated problem dynamics, we construct an exact cost Hamiltonian for Max-E3-SAT using a Higher-Order Unconstrained Binary Optimization (HUBO) formulation similar to [39]. We map the classical binary variables $x_i \in \{0,1\}$ to Pauli-Z operators via the transformation $x_i = \frac{I - Z_i}{2}$, where the computational basis state $|0\rangle$ corresponds to a logical 0. For each clause C_r, we construct a penalty term H_{C_r} by taking the product of the literal penalties. If the j-th literal in the clause is a positive variable x_i, its penalty is $(1 - x_i)$; if it is a negative literal

$\neg x_i$, its penalty is x_i. Mapping these to Pauli-Z operators, the clause Hamiltonian becomes:

$$H_{C_r} = \prod_{j=1}^{3} P_{r,j}, \quad \text{where } P_{r,j} = \begin{cases} \frac{I+Z_i}{2} & \text{if } l_{r,j} = x_i \\ \frac{I-Z_i}{2} & \text{if } l_{r,j} = \neg x_i. \end{cases} \tag{1}$$

The total cost Hamiltonian is constructed by summing these terms over all m clauses: $H_C = \sum_{r=1}^{m} H_{C_r}$. Consequently, the expectation value $\langle H_C \rangle$ corresponds directly to the expected number of violated clauses, and the ground state energy E_0 represents the global minimum of unsatisfied clauses.

C. Operator Pool for Max-E3-SAT

The ADAPT-QAOA framework requires an operator pool $\mathcal{P}$ from which to choose operators to form each mixer layer of the ansatz. The original ADAPT-QAOA pool included only operators that contained an even number of Y or Z Paulis as the problem exhibits bit-flip symmetry. The native formulation of the Max-E3-SAT problem does not exhibit this bit-flip symmetry. We construct an expanded pool containing all combinations of the standard QAOA mixer, and single and two-qubit Pauli operators. Single qubit Z and two-qubit ZZ operators are excluded from the pool, as they commute with the Cost Hamiltonian H_C comprising of only Z, ZZ and ZZZ type operators. Our complete operator pool $\mathcal{P}$ is therefore defined as:

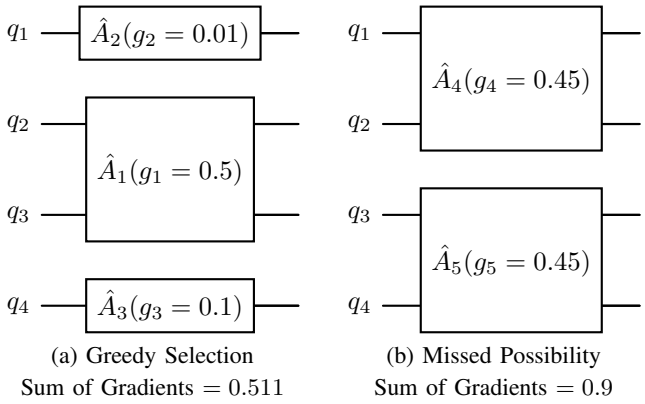
$$\begin{aligned} \mathcal{P} = &\left\{ \sum_{k=1}^{n} X_k \right\} \cup \{X_i, Y_i\}_{\forall i} \\ &\cup \{X_i X_j, Y_i Y_j, X_i Y_j, X_i Z_j, Y_i Z_j\}_{\forall i,j,\, i \neq j}. \end{aligned} \tag{2}$$

D. MosaicADAPT-QAOA for High Quality Supervision

We observe that the operator selection method in TETRIS-ADAPT-VQE is based on a greedy strategy, as it chooses the operator with the current best gradient magnitude. It is susceptible to cases where selecting the current best operator prevents the inclusion of multiple near-best operators that in combination could have provided a greater total energy reduction. See Figure 1 for a representative example.

We propose a selection criterion that aims to maximize the global sum of gradient magnitudes for all selected operators at each QAOA layer. Formally, we model disjoint operator selection at a particular layer as a Maximum Weight Independent Set (MWIS) problem on an incompatibility graph. In such a graph, the nodes represent operators, the node weights are kept proportional to their respective gradient magnitudes, and the edges denote overlapping qubit supports (Figure 2). Solving MWIS on this graph will yield a disjoint set of operators whose sum of gradients is maximized. Since MWIS is NP-hard, we employ an approximate classical approach KaMIS [40]. Specifically, we utilize the Memetic MWIS solver [41], along with its internal scalable kernelization routines [42], to maximize the total gradient sum per layer.

Given an operator pool $\mathcal{P}$, cost Hamiltonian $\hat{H}_C$, initial cost-evolution angle γ_0, gradient threshold ϵ_g, and maximum number of adaptive layers $L_{\max}$, MosaicADAPT-QAOA initializes



(a) Greedy Selection
Sum of Gradients $= 0.511$

(b) Missed Possibility
Sum of Gradients $= 0.9$

Fig. 1: A Representative Conflict Illustration. Operator generators $\hat{A}_i$ are labeled with their corresponding gradient magnitudes in parentheses. In (a), selecting the highest individual gradient ($\hat{A}_1$) blocks the selection possibility of $\hat{A}_4$ and $\hat{A}_5$, leading to a lower total sum of gradients as compared to (b).

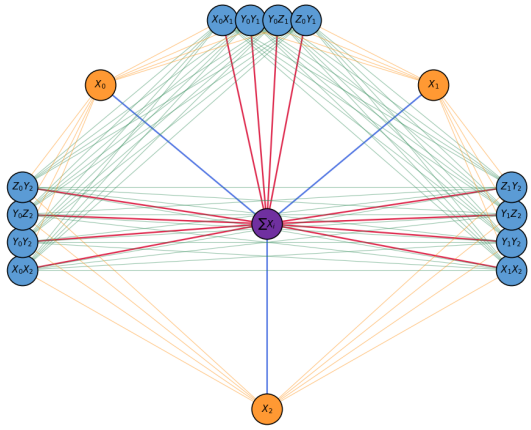


Fig. 2: Incompatibility graph representation for a 3-qubit problem instance, used for operator selection. Each node in the graph represents a candidate unitary operator from the pool $\mathcal{P}$, An edge connects two nodes if and only if their corresponding operators have overlapping qubit support. Sets of overlapping two-qubit operators form fully connected cliques and are omitted to maintain legibility.

the state in the uniform superposition and iteratively adds one mixer layer at a time. At each iteration k, the previous state $|\psi^{(k-1)}\rangle$ is first evolved by $e^{-i\gamma_0 \hat{H}_C}$ for gradient evaluation. For every candidate operator $\hat{A}_j \in \mathcal{P}$, we compute the gradient score $g_j = -i\langle\psi^{(k-1)}|e^{-i\gamma_0\hat{H}_C}[\hat{H}_C, \hat{A}_j]e^{-i\gamma_0\hat{H}_C}|\psi^{(k-1)}\rangle$. The candidate operators are then represented as an incompatibility graph whose edges connect operators with overlapping qubit support. Solving the resulting MWIS instance gives the disjoint operator set $\mathcal{A}_{\text{layer}}$ added at the next mixer layer. After each addition, all variational parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ are reoptimized with BFGS [43]. The procedure stops when all gradient scores fall below ϵ_g or when $L_{\max}$ layers have been added. The

overall MosaicADAPT-QAOA procedure is described at a high level in Algorithm 1.

Algorithm 1 MosaicADAPT-QAOA

Require: $\mathcal{P}$, $\hat{H}_C$, γ_0, ϵ_g, $L_{\max}$
Ensure: Optimized parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ and final state $|\psi\rangle$
$k \leftarrow 0$
$|\psi^{(0)}\rangle \leftarrow |+\rangle^{\otimes n}$; $\boldsymbol{\gamma}, \boldsymbol{\beta} \leftarrow \emptyset$
while $k < L_{\max}$ **do**
  for $\hat{A}_j \in \mathcal{P}$ **do**
    $g_j = -i\langle\psi^{(k-1)}|e^{-i\gamma_0\hat{H}_C}[\hat{H}_C, \hat{A}_j]e^{-i\gamma_0\hat{H}_C}|\psi^{(k-1)}\rangle$
  end for
  if $\max_j \|g_j\| < \epsilon_g$ **then**
    break
  end if
  Construct $G = (V, E)$ with $V = \mathcal{P}$ and $(u, v) \in E$ iff $\mathrm{supp}(\hat{A}_u) \cap \mathrm{supp}(\hat{A}_v) \neq \emptyset$ I
  $\mathcal{A}_{\text{layer}} \leftarrow \mathrm{MWIS}(G, \boldsymbol{g})$
  Add mixer layer $\mathcal{A}_{\text{layer}}$ to the ansatz
  $(\boldsymbol{\gamma}, \boldsymbol{\beta}) \leftarrow \arg\min_{\boldsymbol{\gamma},\boldsymbol{\beta}} \langle\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})|\hat{H}_C|\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle$
  $|\psi^{(k)}\rangle \leftarrow |\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle$
  $k \leftarrow k + 1$
end while
return $(\boldsymbol{\gamma}, \boldsymbol{\beta})$, $|\psi\rangle$

E. Q3SAT-GPT for Circuit Generation

We detail a QAOA-GPT [28] architecture that generates optimized circuits to bypass the classical optimization overhead. The input to this model is a 3-CNF formula and the output is a tokenized representation of a MosaicADAPT-QAOA circuit.

Formula Canonicalization. Aiming to help the model learn similarities between instances, we map the 3-CNF formulas $\mathcal{F}$ to a unique string representation. To achieve this, we apply a deterministic lexicographic sorting to all instances in the dataset. The canonicalization process is defined by two strict ordering rules: (1) *Intra-Clause Literal Ordering:* Within each clause C_r, literals $l_{r,j}$ are sorted based on a strict total order $<_L$. The sort key is the variable index i. Thus, the literal ordering follows: $x_1 <_L x_2 <_L x_3 <_L \cdots <_L x_n$. (2) *Inter-Clause Ordering:* Once the literals within all m clauses are internally sorted, the clauses themselves are sorted lexicographically based on their ordered literals. If two variables have the same index and are of opposite polarity, we consider $x_i <_L \neg x_i$. For example, given an unsorted formula $\mathcal{F}$:

$$(x_2 \vee \neg x_4 \vee x_3) \wedge (x_1 \vee x_3 \vee x_2) \wedge (x_1 \vee x_2 \vee \neg x_4),$$

we fist sort intra-clause literals:

$$(x_2 \vee x_3 \vee \neg x_4) \wedge (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_4).$$

Then, the inter-clause sorting orders the clauses lexicographically to produce the final canonical string fed to the model:

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_4) \wedge (x_2 \vee x_3 \vee \neg x_4).$$

Tokenization. We adapt the tokenization scheme from QAOA-GPT [28] to 3-CNF formulas and MosaicADAPT-QAOA circuits. The input sequence contains the canonicalized formula, using tokens x_i and $\neg x_i$ for literals and | as the clause separator, together with the standard $\langle bos\rangle$, $\langle end\_of\_formula\rangle$, and $\langle eos\rangle$ tokens. To condition the model on global formula structure, we construct a Literal-Clause Graph (LCG) with $2n+m$ nodes, corresponding to the literals and clauses, and connect each literal to the clauses in which it appears. FEATHER embeddings of this graph are projected through an MLP and added to the token and positional embeddings. The output sequence follows the QAOA-GPT circuit-token format, except that each MosaicADAPT-QAOA layer contains all disjoint operators selected for that layer, each paired with its corresponding β parameter, followed by the layer's γ parameter. Thus, a layer is represented as $\langle new\_layer\_p\rangle, \mathrm{op}_1, \beta_1, \ldots, \mathrm{op}_r, \beta_r, \gamma$. See Figure 3 for a visual representation.

Loss Function. To ensure that the model is trained only on predicting valid and optimized circuit sequences, we modify the Categorical Cross Entropy loss function used in QAOA-GPT to add the loss only for the output tokens as:

$$L = -\sum_t \mathbb{1}_{\{y_t \neq 0\}} \log(p(y_t|X_{\leq t})), \tag{3}$$

where $\mathbb{1}_{\{y_t \neq 0\}}$ is an indicator function that masks the loss for parameters associated with the input formula and padding tokens (target index 0). In contrast to the original QAOA-GPT model, we do not add a sliding window to extract subsequences, but rather use a single context window of 1024 tokens, ensuring that all the sequences in the dataset fit in the context window, including the input formula, the output circuit and the separator tokens. The tokens after the $\langle eos\rangle$ token are padded with $\langle pad\rangle$ tokens, and the loss is masked for the $\langle pad\rangle$ tokens.

Inference and Circuit Decoding. During inference, Q3SAT-GPT generates optimized quantum circuits in a single forward pass. Given a tokenized formula, the model predicts a sequence of tokens autoregressively. At each step, a token is sampled from the output distribution using a temperature T to control diversity. The predicted tokens are mapped back to quantum gates (operators) and their corresponding variational parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$. We consider a circuit sequence as the tokens between the $\langle end\_of\_formula\rangle$ and the $\langle eos\rangle$ tokens.

IV. Experimental Setup

A. Evaluation Metrics

Approximation Ratio. The quality of the results is measured in terms of Approximation Ratio (AR), where the AR of a solution for a formula is defined as

$$\mathrm{AR} = \frac{\text{Number of clauses satisfied by a candidate method}}{\text{Maximum number of clauses that can be satisfied}}$$

The maximum number of clauses that are satisfiable was computed using Gurobi [44].

Error Rate. The structural validity of the GPT generated circuits is measured using Error Rate (ER). For each problem instance, the GPT model can generate different circuits based on its temperature parameter. We sample 5 circuits from the model for each formula. For generated-circuit results, AR is

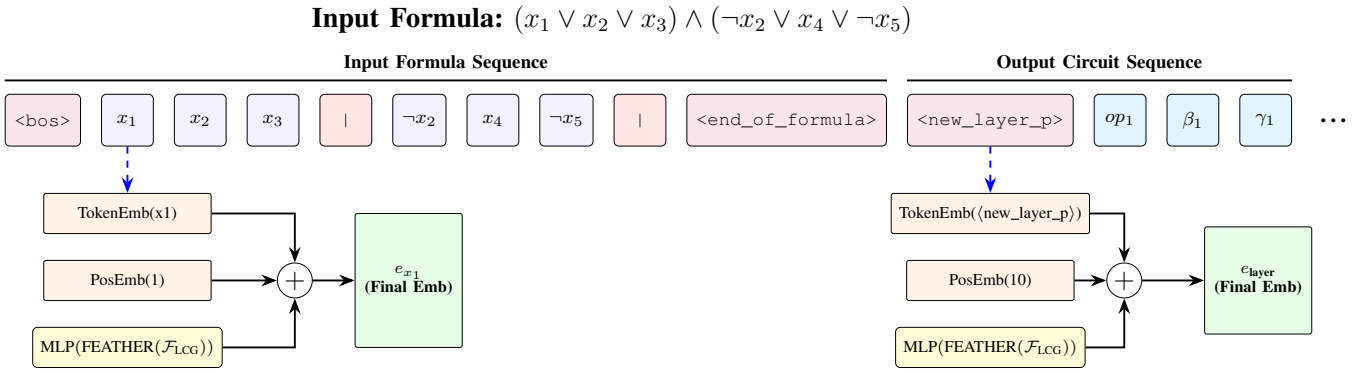


Fig. 3: Detailed pipeline demonstrating formatting, tokenization, and embeddings. The | delimiter is used to segment clauses, with explicit OR ($\vee$) tokens omitted.

computed over valid generated circuits, while ER is reported at both the formula and circuit levels. The Formula Error Rate denotes the percentage of formulas in which no sampled circuit was valid for the formula. The Circuit Error Rate denotes the total percentage of sampled circuits that were invalid.

B. Setup for MosaicADAPT-QAOA

Baselines for Adaptive Circuit Construction. To evaluate the efficacy of MosaicADAPT-QAOA, we compare it against two primary adaptive baselines: (1) *ADAPT-QAOA*, which selects a single operator per layer based on the highest individual gradient, and (2) *TETRIS-QAOA*, which is a modification of ADAPT-QAOA and employs a greedy selection of multiple disjoint operators. All three methods use the same operator pool defined in Section III-C and the same hyperparameters[1]. The algorithm's termination is primarily dictated by two dominant criteria: the convergence of all operator gradients below 10^{-6} or the reaching of the 20-layer maximum adaptation limit.

Benchmark Test Set. We generated a benchmark set of 100 Max-E3-SAT instances. Each instance contains 10 variables and is satisfiable. 50 of the 100 instances are generated using the Uniform Random generation process, and the other 50 instances are from the Balanced distribution [32] generated using the `satqubolib` Python library [45].

For balanced instances, the number of clauses m for an instance with n (here $n = 10$) variables is drawn from a randomized ratio around the phase transition value $\alpha_0 = 3.6$:

$$\alpha = \alpha_0 (1 + \delta), \qquad \delta \sim \text{Uniform}(-0.2,\ 0.2), \qquad (4)$$

so that $\alpha \in [2.88,\ 4.32]$ (i.e. $\alpha_0 \pm 20\%$). For the uniform random instances, the number of clauses m for an instance with n variables is drawn from a uniform distribution between $\alpha = 4.26 \pm 20\%$.

C. Setup for Q3SAT-GPT

Dataset Construction. We conduct our Q3SAT-GPT training on two problem sizes: 10- and 12-variable SAT instances. For each problem size, we generate a dataset containing 50000 Max-E3-SAT instances. It consists of 25K uniform random instances and 25K balanced instances. For each type, half of the instances are satisfiable and half are unsatisfiable. We ensure that each instance is unique after canonicalization. The dataset finally is split into train (80%), validation (10%), and test (10%) partitions stratified by satisfiability and formula generation method.

Model Variants. We train a total of 4 models[1] each, for 10-variable and 12-variable Max-E3-SAT:

1) Without LCG-graph embeddings: A pure transformer model that generates circuit parameters conditioned exclusively on the literal tokenized 3-CNF formula.
2) With FEATHER embeddings on LCG-graph embeddings: The FEATHER embeddings are projected to the input embedding layer through a multi-layer perceptron, aiming to provide global context about the formula. These embeddings are added to the token and positional embeddings.

For each of the above variant, we train two models based on the initial gamma γ_0 hyperparameter:

1) $\gamma_0 = 0.5$: An initial gamma of 0.5 is used for producing all 50K optimized circuits using MosaicADAPT-QAOA.
2) $\gamma_0 = \text{argmax}_{\gamma \in \{0.01, 0.1, 0.5\}} \text{AR}(\gamma)$: A grid search over different initial gamma values is performed for all 50K instances. The circuit yielding the highest AR is selected, with ties broken by fewer layers.

Model Selection. We perform model selection on the AR metric using a fixed validation set. 50 validation set formulas are randomly sampled from the validation partition at startup with a fixed seed, held fixed for the entire training run. For each instance, Q3SAT-GPT generates 5 circuits autoregressively. The generated circuits are then evaluated on the AR and ER metrics. The model with the highest AR and lowest ER is selected for deployment on the test set.

V. Results

A. MosaicADAPT-QAOA

We compare the performance of the three methods (ADAPT-QAOA, TETRIS-QAOA, MosaicADAPT-QAOA) at a low

[1]For full details about ADAPT variants' hyperparameters and Q3SAT-GPT model architecture, see our extended paper at https://arxiv.org/abs/2604.27324

initial gamma value, i.e., $\gamma_0 = 0.01$, and a relatively higher initial gamma value, i.e., $\gamma_0 = 0.5$, to understand the behavior of the algorithms. Overall, we observe that MosaicADAPT-QAOA performs better in terms of clause satisfaction rate and circuit depth than TETRIS-QAOA and ADAPT-QAOA. Table I shows that the mean and median clause satisfaction rates for MosaicADAPT-QAOA are higher than those of TETRIS-QAOA and ADAPT-QAOA. At any particular layer, the AR of MosaicADAPT-QAOA is higher on average (Figure 4).

TABLE I: Comparison of Performance on Different ADAPT variants. The Median number of layers to reach 99.9% AR are shown only for the instances that reached this target within 20 layers.

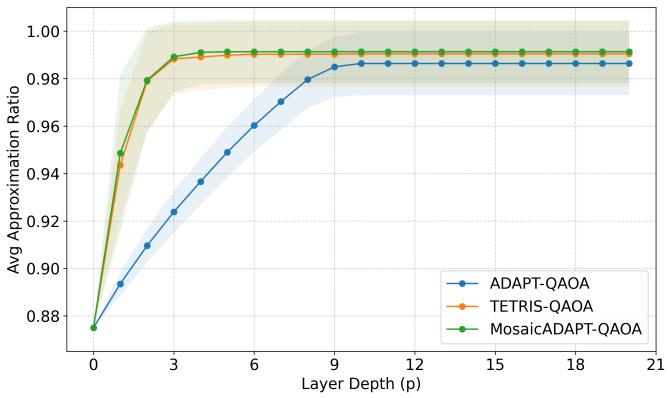
Method	Mean AR (%)	Median AR (%)	Median Layers to 99.9% AR	Fraction of Stuck Instances
		$\gamma_0 = 0.01$		
ADAPT-QAOA	98.64	98.62	9	13/100
TETRIS-QAOA	99.05	**100.00**	**2**	23/100
MosaicADAPT-QAOA	**99.13**	**100.00**	**2**	27/100
		$\gamma_0 = 0.5$		
ADAPT-QAOA	99.68	99.88	16	0/100
TETRIS-QAOA	99.68	99.88	16	0/100
MosaicADAPT-QAOA	**99.71**	**100.00**	**12**	0/100

Sensitivity to Initial Gamma. For low initial gamma values, we observe that all 3 methods are prone to early stopping due to a sudden drop in gradients below the accepted threshold (Figure 5). We define an instance as "stuck" if in 1000 shot sampling, the energies of all final bitstrings are strictly greater than the ground state energy. Table I shows that $\gamma_0 = 0.01$ led to more frequent such cases where instances were stuck due to early stopping. Importantly, in cases where the tendency of getting stuck is less ($\gamma_0 = 0.5$), MosaicADAPT-QAOA required a median of 4 fewer layers than the other variants.

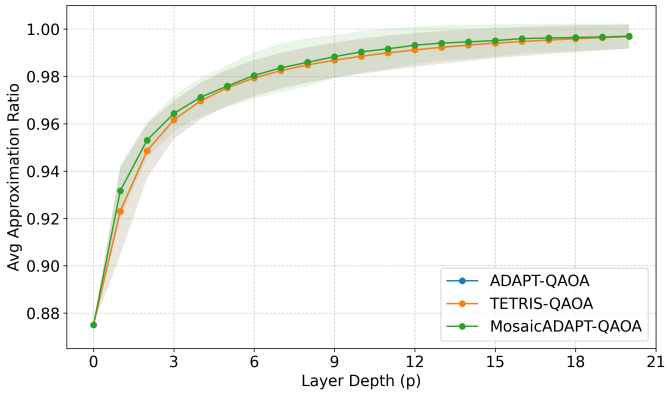
B. Q3SAT-GPT Results

Performance on Test Instances. Table II presents the results on the Approximation Ratio and Error Rate metrics. We observe that, in general, the circuits generated with FEATHER embeddings have a lower error rate, though the difference in performance in terms of AR in inconclusive. The AR achieved on balanced instances is consistently higher than that on random instances, suggesting that the model is able to differentiate between the types of instances and that there is a structure to the resultant circuits for the balanced instances. In addition, the model performed better when trained on a dataset optimized with a single $\gamma_0 = 0.5$ rather than a grid search.

Inference Time Comparison. Table III shows the amortized inference time for Q3SAT-GPT predicted circuits and compares it against the time required to find a single MosaicADAPT-QAOA circuit on a single-threaded CPU. Q3SAT-GPT performance is reported as amortized inference time, calculated by dividing the total batch execution time by the batch size ($B = 128$).



(a) $\gamma_0 = 0.01$



(b) $\gamma_0 = 0.5$

Fig. 4: Comparison of the energy convergence across adaptations. Note that in Figure (b), plots of TETRIS-QAOA and ADAPT-QAOA overlap due to the same operators being chosen by both the methods (see Figure 6).

VI. Discussion

A. Circuit Structure Sensitivity to Initial Gamma

We often observe that for low initial values of gamma ($\gamma_0 < 0.1$), the classical optimizer pushes the gamma values to be close to 0. It also pushes the beta values to $\pm\frac{\pi}{4}$. This is in line with recent studies that observe the same effect, and that ADAPT-QAOA is prone to get stuck in an excited state [46]. Additionally, at low initial γ values, all three methods mostly choose Y-type operators or YZ-type operators (Figure 6). With the corresponding β value of $\pm\frac{\pi}{4}$, these operators make a qubit go from a uniform superposition of $|0\rangle$ and $|1\rangle$ to either $|0\rangle$ or $|1\rangle$. Since the γ values are pushed to 0, the behavior of the optimized circuit does not seem to leverage the phase changes induced by the cost Hamiltonian in QAOA. Since the circuits optimized on low initial γ values had final β values close to $\pm\frac{\pi}{4}$ and final gamma values close to 0, Q3SAT-GPT does not seem to yet find out which qubit in a uniform superposition should be

TABLE II: Comparison of Model Training Methodologies

N	Selected Instance	Distribution	Formula ER (%)		Circuit ER (%)		Best AR (%)		Avg AR (%)		MosaicADAPT QAOA AR (%)
			F	W/O	F	W/O	F	W/O	F	W/O	
10	Grid Search	Balanced	0.00	0.00	5.00	4.14	98.41	98.52	97.02	97.36	99.83
		Random	0.00	0.00	0.60	0.12	95.17	96.03	93.85	94.42	99.96
	$\gamma_0 = 0.5$	Balanced	0.00	0.00	0.05	1.16	98.88	99.00	98.79	98.83	99.42
		Random	0.76	6.56	9.82	16.10	97.83	92.60	96.39	91.37	99.84
12	Grid Search	Balanced	0.00	0.00	0.53	0.45	97.04	98.55	94.18	98.21	99.77
		Random	0.00	0.00	0.61	1.66	93.98	95.34	92.41	94.18	99.96
	$\gamma_0 = 0.5$	Balanced	0.00	0.00	0.01	0.01	97.28	98.62	96.94	98.51	99.34
		Random	0.00	0.00	1.30	5.30	96.53	97.81	94.89	95.90	99.85

N: Number of variables in MAX-E3-SAT, **ER**: Error Rate, **AR**: Approximation Ratio , **F**: With FEATHER, **W/O**: Without FEATHER. AR values are calculated only for valid circuits and are expressed as a percentage

TABLE III: Runtime Performance Comparison.

Method	N=10 (s)	N=12 (s)
Amortized Inference (GPU)		
Q3SAT-GPT	0.16	0.45
Sequential Optimization (CPU)		
MosaicADAPT-QAOA	88.21	503.72

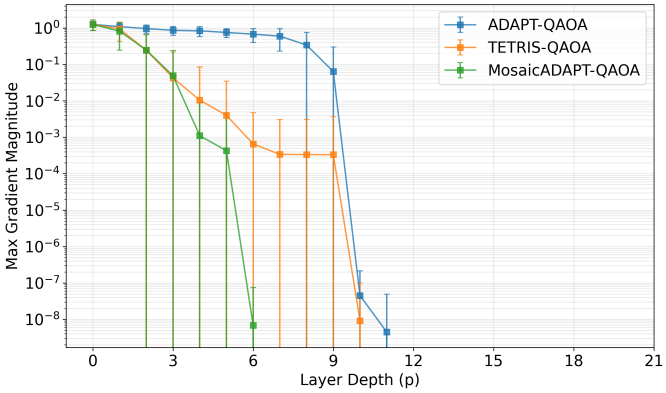
pushed to a computational basis state. This might explain that the performance of Q3SAT-GPT with a grid-search could be improved. On the other hand, at higher initial γ_0 values, we observe that ADAPT-QAOA and TETRIS-QAOA mostly choose the standard QAOA mixer in the initial mixer layers. However, MosaicADAPT-QAOA chooses the single-qubit X-operators and essentially moves towards multi-angle QAOA in the initial layers. As the algorithm progresses, it makes use of the other operators in the pool in the subsequent layers. This allows MosaicADAPT-QAOA to perform better than ADAPT-QAOA and TETRIS-QAOA at $\gamma_0 = 0.5$.

B. Depth vs. Parameter Tradeoff

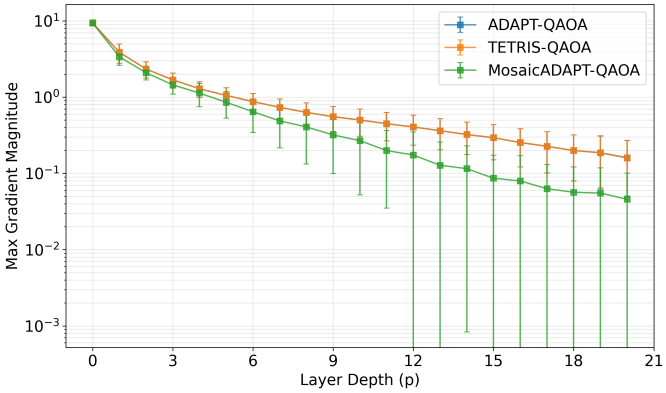
We distinguish between adaptive layers, selected mixer operators, and variational parameters. MosaicADAPT-QAOA reduces the number of adaptive layers by selecting denser compatible operator sets, but this can increase the number of trainable parameters, especially in the $\gamma_0 = 0.5$ regime where ADAPT-QAOA and TETRIS-QAOA often select the standard QAOA mixer in the initial layers. While ADAPT-QAOA and TETRIS-QAOA required a median number of parameters of 17 and 17.5 respectively, the MosaicADAPT-QAOA method required a median of 97 parameters to reach the 99.9% AR. This motivates the use of generative AI based approaches. If the GPT model is able to learn the parameters, the massive parameter optimization overhead is mitigated, while still learning from a rich dataset.

C. Q3SAT-GPT Predicted Circuit Structure

Figure 7 compares the variance of γ and β values for circuits generated by Q3SAT-GPT and circuits optimized directly by MosaicADAPT-QAOA. For balanced instances, the model

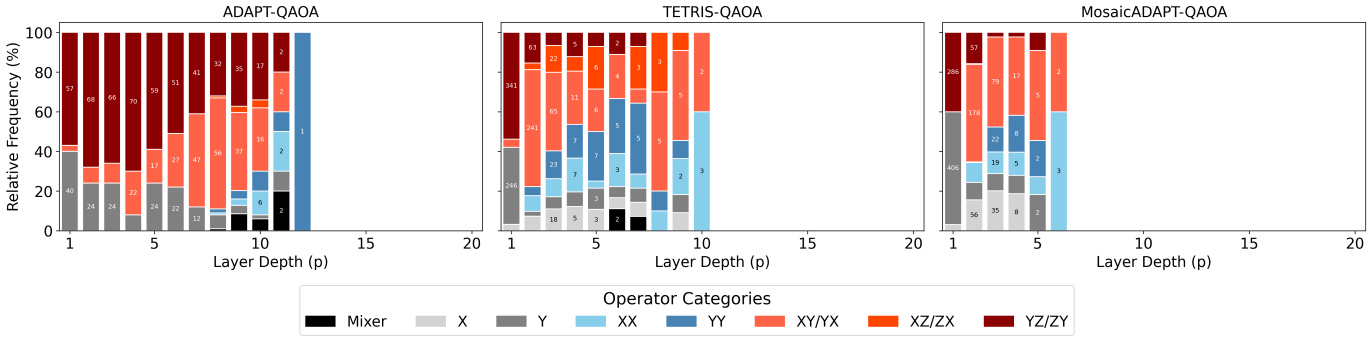


(a) $\gamma_0 = 0.01$

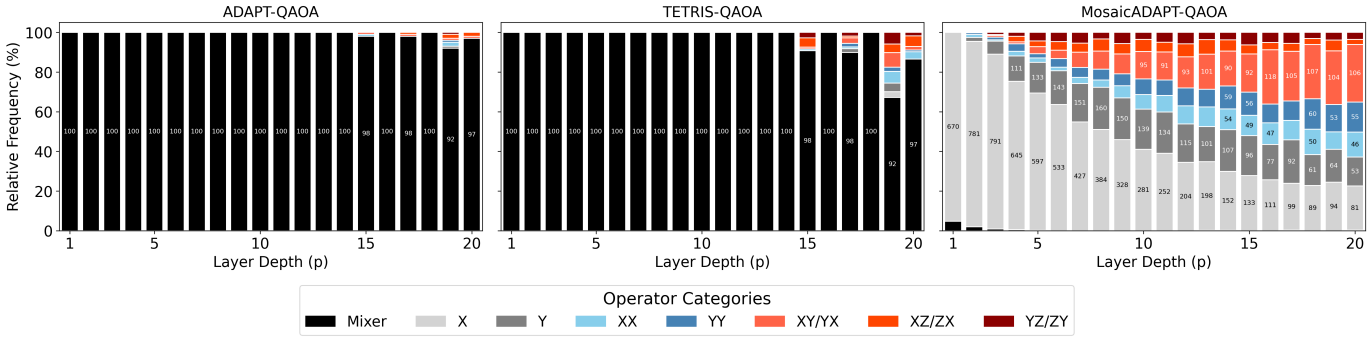


(b) $\gamma_0 = 0.5$

Fig. 5: Comparison of the maximum gradient norm across all operators in the mixer pool. Note that in Figure (b), plots of TETRIS-QAOA and ADAPT-QAOA overlap due to the same operators having the maximum gradient for the most part (see Figure 6).



(a) $\gamma_0 = 0.01$



(b) $\gamma_0 = 0.5$

Fig. 6: Evolution of the operator selection across adaptations. Note that the count inside a bar is that of the total number of operators selected in that layer across all instances.

often predicts highly concentrated parameter values, yet these circuits still reach a 98.51% approximation ratio. For random instances, the generated circuits show more variability, and in a few cases the model does not predict the $\langle eos \rangle$ token by the 20th layer. Another issue is that the model also does not consistently reproduce the late-layer operators selected by MosaicADAPT-QAOA. This does not always harm solution quality because the corresponding late-layer β values are often near zero. However, generating higher-quality circuits will likely require better recovery of the operators selected in the final adaptive layers.

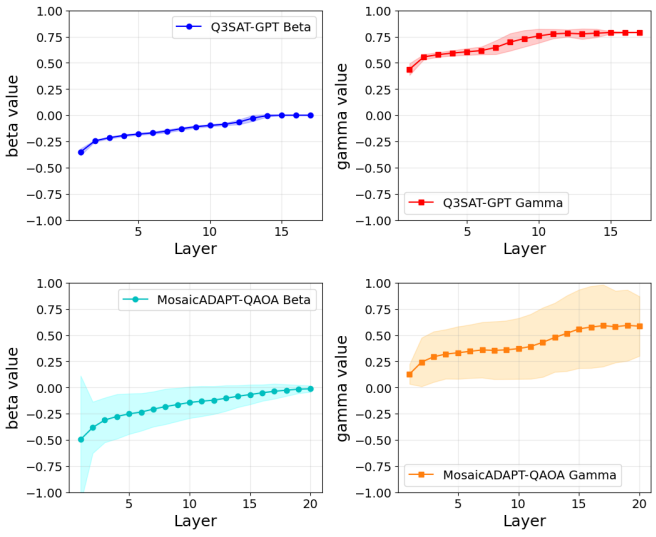
D. Lessons Learned and Future Directions

The generated circuits suggest that the structure of the MosaicADAPT-QAOA training data strongly shapes the behavior of Q3SAT-GPT. In the training set, the first few layers are often dominated by single-qubit Pauli-X operators, while operators that appear in later adaptive layers are much less frequent. As a result, a model trained with categorical cross-entropy can overrepresent early-layer patterns when generating later layers. This suggests that future models should treat layer position and operator diversity more explicitly, for example through layer-aware losses, reweighting of underrepresented late-layer operators, or targeted augmentation of training sequences.

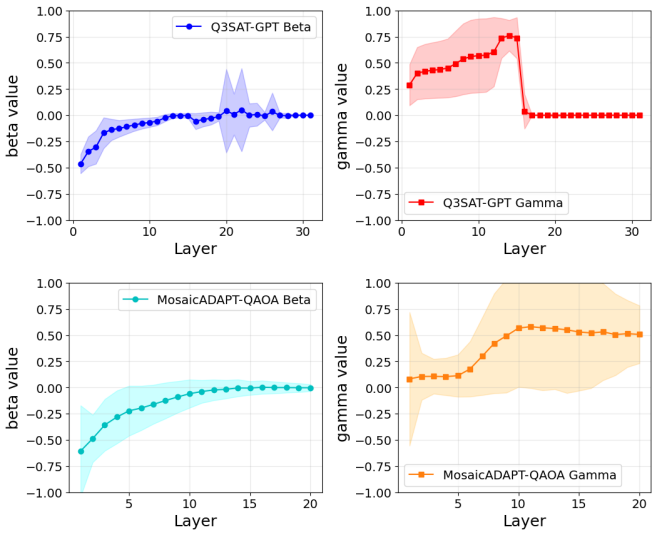
Syntactic validity and circuit quality should also be separated more clearly. The current model can occasionally generate structurally invalid circuits, especially at larger depths where the output sequence is longer and the set of valid next tokens is more constrained. Constrained decoding would allow the model to sample only structurally valid operators, parameters, and layer delimiters at each generation step. These constraints would not guarantee high approximation ratio by themselves, but they would remove avoidable structural errors and let model capacity focus on circuit quality rather than circuit grammar. Repetition penalties or diversity-promoting decoding rules may also help prevent the model from repeatedly selecting the same dominant operator types in later layers.

The performance gap between balanced and random instances also suggests that richer instance-level information remains important. Q3SAT-GPT appears to learn circuit patterns more easily when formulas have more regular structure. SAT-specific neural architectures such as NeuroSAT [47] or SAT-GATv2 [48] could provide stronger inductive bias than the current literal-clause graph embeddings alone, as related graph-based embeddings have helped in QAOA parameter learning [17]. Such embeddings may help the model distinguish which adaptive operators are useful for a particular formula, rather than relying primarily on global dataset-level circuit patterns.

Finally, Q3SAT-GPT should be viewed as a first step towards closed-loop generative quantum circuit discovery. The present model learns from offline MosaicADAPT-QAOA trajectories and generates circuits without classical variational op-



(a) Balanced instances.



(b) Random instances.

Fig. 7: Comparison of Q3SAT-GPT predicted parameters against MosaicADAPT-QAOA optimized parameters on the test set. Solid lines denote the mean parameter value across samples, and shaded regions denote ± 1 standard deviation. This plot is for the configuration containing 12 qubits, with $\gamma_0 = 0.5$, and utilizes FEATHER embeddings. Since there are multiple beta values at a particular layer for MosaicADAPT-QAOA, all beta values are averaged at that layer.

timization. A natural next step is to add feedback from circuit execution, either through simulation or quantum hardware, so that generated circuits can be reinforced according to measured solution quality. Such a feedback loop could combine the speed of autoregressive generation with quality signals from actual circuit performance. Finding a balance between them that would keep the method scalable is a challenging future direction.

Acknowledgments

This research was supported in part through the use of DARWIN computing system funded by NSF Grant #1919839. K. Shirali was supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, Co-design Center for Quantum Advantage (C2QA) under Contract Number DE-SC0012704. This research was supported in part by NSF Grant #2427042. We acknowledge the use of Generative AI tools (Cursor, Google Antigravity) to speed up some coding tasks. All generated code was thoroughly reviewed and unit-tested wherever applicable.

References

[1] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, "Strengths and weaknesses of quantum computing," *SIAM journal on Computing*, vol. 26, no. 5, pp. 1510–1523, 1997.

[2] L. Fortnow and J. Rogers, "Complexity limitations on quantum computation," *Journal of Computer and System Sciences*, vol. 59, no. 2, pp. 240–252, 1999.

[3] S. Aaronson, "How much structure is needed for huge quantum speedups?" *arXiv preprint arXiv:2209.06930*, 2022.

[4] P. Lougovski, O. Parekh, J. Broz, M. Byrd, J. C. Chapman, Y. Chembo, W. A. de Jong, E. Figueroa, T. S. Humble, J. Larson *et al.*, "Report for the ASCR Workshop on Basic Research Needs in Quantum Computing and Networking-2023," USDOE Office of Science (SC)(United States), Tech. Rep., 2023.

[5] M. E. Beverland, P. Murali, M. Troyer, K. M. Svore, T. Hoefler, V. Kliuchnikov, G. H. Low, M. Soeken, A. Sundaram, and A. Vaschillo, "Assessing requirements to scale to practical quantum advantage," *arXiv preprint arXiv:2211.07629*, 2022.

[6] A. Katabarwa, K. Gratsea, A. Caesura, and P. D. Johnson, "Early fault-tolerant quantum computing," *PRX quantum*, vol. 5, no. 2, p. 020101, 2024.

[7] D. Herman, C. Googin, X. Liu, Y. Sun, A. Galda, I. Safro, M. Pistoia, and Y. Alexeev, "Quantum computing for finance," *Nature Reviews Physics*, vol. 5, no. 8, p. 450–465, Jul. 2023. [Online]. Available: http://dx.doi.org/10.1038/s42254-023-00603-1

[8] R. Shaydulin, H. Ushijima-Mwesigwa, I. Safro, S. Mniszewski, and Y. Alexeev, "Network community detection on small quantum computers," *Advanced Quantum Technologies*, vol. 2, no. 9, p. 1900029, 2019. [Online]. Available: https://dx.doi.org/10.1002/qute.201900029

[9] A. Fedorov and M. Gelfand, "Towards practical applications in quantum computational biology," *Nature Computational Science*, vol. 1, no. 2, pp. 114–119, 2021.

[10] Y. Alexeev, M. H. Farag, T. L. Patti, M. E. Wolf, N. Ares, A. Aspuru-Guzik, S. C. Benjamin, Z. Cai, S. Cao, C. Chamberland *et al.*, "Artificial intelligence for quantum computing," *Nature Communications*, vol. 16, no. 1, p. 10829, 2025.

[11] J. Merilehto, "Generative ai for quantum circuits and quantum code: A technical review and taxonomy," *arXiv preprint arXiv:2603.16216*, 2026.

[12] J. Håstad, "Some optimal inapproximability results," *Journal of the ACM (JACM)*, vol. 48, no. 4, pp. 798–859, 2001.

[13] H. van Maaren and J. Franco, "SAT Competitions," https://satcompetition.github.io/.

[14] M. Järvisalo, D. Le Berre, O. Roussel, and L. Simon, "The international sat solver competitions," *AI Magazine*, vol. 33, no. 1, pp. 89–92, 2012.

[15] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," *arXiv preprint arXiv:1411.4028*, 2014. [Online]. Available: https://arxiv.org/abs/arXiv:1411.4028

[16] A. Abbas, A. Ambainis, B. Augustino, A. Bärtschi, H. Buhrman, C. Coffrin, G. Cortiana, V. Dunjko, D. J. Egger, B. G. Elmegreen *et al.*, "Challenges and opportunities in quantum optimization," *Nature Reviews Physics*, pp. 1–18, 2024.

[17] J. Falla, Q. Langfitt, Y. Alexeev, and I. Safro, "Graph representation learning for parameter transferability in quantum approximate optimization algorithm," *Quantum Machine Intelligence*, vol. 6, no. 2, p. 46, 2024.

[18] F. G. Brandao, M. Broughton, E. Farhi, S. Gutmann, and H. Neven, "For fixed control parameters the quantum approximate optimization algorithm's objective function value concentrates for typical instances," *arXiv preprint arXiv:1812.04170*, 2018.

[19] A. Galda, X. Liu, D. Lykov, Y. Alexeev, and I. Safro, "Transferability of optimal qaoa parameters between random graphs," in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 2021, pp. 171–180.

[20] D. J. Egger, J. Mareček, and S. Woerner, "Warm-starting quantum optimization," *Quantum*, vol. 5, p. 479, 2021.

[21] R. Herrman, P. C. Lotshaw, J. Ostrowski, T. S. Humble, and G. Siopsis, "Multi-angle quantum approximate optimization algorithm," *Scientific Reports*, vol. 12, no. 1, p. 6781, 2022.

[22] X. Liu, R. Shaydulin, and I. Safro, "Quantum approximate optimization algorithm with sparsified phase operator," in *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 2022, pp. 133–141.

[23] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, "Hybrid quantum-classical algorithms for approximate graph coloring," *Quantum*, vol. 6, p. 678, 2022.

[24] L. Zhu, H. L. Tang, G. S. Barron, F. A. Calderon-Vargas, N. J. Mayhall, E. Barnes, and S. E. Economou, "Adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer," *Phys. Rev. Res.*, vol. 4, p. 033029, Jul 2022. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevResearch.4.033029

[25] X. Liu, A. Angone, R. Shaydulin, I. Safro, Y. Alexeev, and L. Cincio, "Layer VQE: A variational approach for combinatorial optimization on noisy quantum computers," *IEEE Transactions on Quantum Engineering*, vol. 3, pp. 1–20, 2022.

[26] A. B. Magann, K. M. Rudinger, M. D. Grace, and M. Sarovar, "Feedback-based quantum optimization," *Physical Review Letters*, vol. 129, no. 25, p. 250502, 2022.

[27] P. G. Anastasiou, Y. Chen, N. J. Mayhall, E. Barnes, and S. E. Economou, "Tetris-adapt-vqe: An adaptive algorithm that yields shallower, denser circuit ansätze," *Physical Review Research*, vol. 6, no. 1, p. 013254, 2024.

[28] I. Tyagin, M. H. Farag, K. Sherbert, K. Shirali, Y. Alexeev, and I. Safro, "QAOA-GPT: Efficient generation of adaptive and regular quantum approximate optimization algorithm circuits," in *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 1. IEEE, 2025, pp. 1505–1515.

[29] D. S. Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer and System Sciences*, vol. 9, no. 3, pp. 256–278, 1974. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022000074800449

[30] C. H. Papadimitriou and M. Yannakakis, "Optimization, approximation, and complexity classes," *Journal of Computer and System Sciences*, vol. 43, no. 3, pp. 425–440, 1991. [Online]. Available: https://www.sciencedirect.com/science/article/pii/002200009190023X

[31] H. H. Hoos and T. Stützle, "Satlib: An online resource for research on sat," *Sat*, vol. 2000, pp. 283–292, 2000.

[32] I. Spence, "Balanced random sat benchmarks," *SAT COMPETITION 2017*, vol. 53, 2017.

[33] D. Mitchell, B. Selman, H. Levesque *et al.*, "Hard and easy distributions of sat problems," in *Aaai*, vol. 92, 1992, pp. 459–465.

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

[35] F. G. S. L. Brandao, M. Broughton, E. Farhi, S. Gutmann, and H. Neven, "For fixed control parameters the quantum approximate optimization algorithm's objective function value concentrates for typical instances," 2018. [Online]. Available: https://arxiv.org/abs/1812.04170

[36] A. Galda, E. Gupta, J. Falla, X. Liu, D. Lykov, Y. Alexeev, and I. Safro, "Similarity-based parameter transferability in the quantum approximate optimization algorithm," *Frontiers in Quantum Science and Technology*, vol. Volume 2 - 2023, 2023. [Online]. Available: https://www.frontiersin.org/journals/quantum-science-and-technology/articles/10.3389/frqst.2023.1200975

[37] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, "An adaptive variational algorithm for exact molecular simulations on a quantum computer," *Nature Communications*, vol. 10, no. 1, Jul. 2019. [Online]. Available: http://dx.doi.org/10.1038/s41467-019-10988-2

[38] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, "Obstacles to variational quantum optimization from symmetry protection," *Physical review letters*, vol. 125, no. 26, p. 260505, 2020.

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

[40] S. Lamm, P. Sanders, C. Schulz, D. Strash, and R. F. Werneck, "Finding near-optimal independent sets at scale," *J. Heuristics*, vol. 23, no. 4, pp. 207–229, 2017. [Online]. Available: https://doi.org/10.1007/s10732-017-9337-x

[41] ernestine großmann, sebastian lamm, christian schulz, and darren strash, "finding near-optimal weight independent sets at scale," in *proceedings of the genetic and evolutionary computation conference, gecco 2023, lisbon, portugal, july 15-19, 2023*, sara silva and luís paquete, Eds. acm, 2023, pp. 293–302. [Online]. Available: https://doi.org/10.1145/3583131.3590353

[42] D. Hespe, C. Schulz, and D. Strash, "Scalable kernelization for maximum independent sets," *ACM Journal of Experimental Algorithmics*, vol. 24, no. 1, pp. 1.16:1–1.16:22, 2019. [Online]. Available: https://doi.org/10.1145/3355502

[43] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer-Verlag, 1999.

[44] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2026. [Online]. Available: https://www.gurobi.com

[45] S. Zielinski, M. Benkard, J. Nüßlein, C. Linnhoff-Popien, and S. Feld, "Satqubolib: A python framework for creating and benchmarking (max-)3sat qubos," in *Innovations for Community Services*, F. Phillipson, G. Eichler, C. Erfurth, and G. Fahrnberger, Eds. Cham: Springer Nature Switzerland, 2024, pp. 48–66.

[46] V. K. Sridhar, Y. Chen, B. Gard, E. Barnes, and S. E. Economou, "Adapt-qaoa with a classically inspired initial state," *arXiv preprint arXiv:2310.09694*, 2023.

[47] D. Selsam, M. Lamm, B. Bünz, P. Liang, L. de Moura, and D. L. Dill, "Learning a sat solver from single-bit supervision," 2019. [Online]. Available: https://arxiv.org/abs/1802.03685

[48] W. Chang and W. Liu, "Sat-gatv2: A dynamic attention-based graph neural network for solving boolean satisfiability problem," *Electronics*, vol. 14, no. 3, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/14/3/423