LEARNING GAUSSIAN BOSON SAMPLING FOR INTEGRATED QUANTUM PHOTONICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Gaussian boson sampling (GBS) is a leading platform for probing the boundary between quantum and classical computation. Given its impact on practical quantum advantage, emulating GBS is crucial, but exact calculation of its output probabilities is computationally expensive and #P-hard in general. Practical GBS workflows repeatedly query related photonic circuits and detection patterns, creating an opportunity to accelerate this computation through machine learning. We formulate GBS probability evaluation as a supervised learning task and introduce **PhOracle**, a physics-informed recurrent model that adapts its matrix computations to the requested detection pattern. We also introduce **GBSBench**, a circuit-disjoint benchmark that evaluates generalization to unseen circuits and unseen detection patterns. PhOracle achieves a median absolute percentage error of **4.56%** on held-out circuit-pattern queries, while matched runtime tests show that PhOracle evaluates probabilities up to **79.8**× faster than the reference solver. We further apply PhOracle to closed-loop quantum circuit optimization. Without task-specific retraining, PhOracle accelerates single-pair and multipair Bell-state circuit discovery by **17.3**× and **36.2**×, respectively, reaching fidelities close to those achieved by reference-solver search. Together, this work provides an empirical basis for future studies of learned probability models across broader quantum-photonic systems and applications.

1 INTRODUCTION

Like classical computers, quantum computers we ultimately hope to build are general-purpose machines. Rather than being limited to a single task, they could run many useful algorithms, including algorithms for cryptanalysis (Shor, 1999), molecular simulation (Aspuru-Guzik et al., 2005), and drug discovery (Santagati et al., 2024). Despite decades of progress, however, building such a machine still requires major advances in both hardware and software, especially in scaling and error correction (Campbell et al., 2017). At present, it is widely believed that there is a formidable challenge in building a practically useful universal quantum computer.

Gaussian boson sampling, or GBS, offers a different path. Although a GBS device is not a universal quantum computer, it can generate photon-count distributions that are considered computationally intractable for classical systems (Hamilton et al., 2017). Theoretical evidence for the hardness of classical simulation makes GBS a leading candidate for demonstrating quantum advantage. Driven by this goal, experimental efforts have focused on demonstrating quantum advantage with GBS hardware (Zhong et al., 2020; Madsen et al., 2022). Meanwhile, theoretical work has advanced both exact and approximate classical simulation of GBS (Oh, 2025; Dodd et al., 2025).

A major obstacle to simulating GBS on classical computers is the exponential growth of exact simulation costs with detected photon number (Quesada & Arrazola, 2020; Quesada et al., 2022). The evaluation of individual output probabilities involves the Hafnian, a combinatorial matrix function whose exact evaluation is #P-hard in general (Hamilton et al., 2017). Despite this cost, simulation is essential both for benchmarking quantum-advantage experiments and for designing and optimizing quantum systems (Bulmer et al., 2022; Hartnett et al., 2026). These needs have driven sustained research on faster exact algorithms and approximations (Quesada et al., 2022; Oh et al., 2024).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

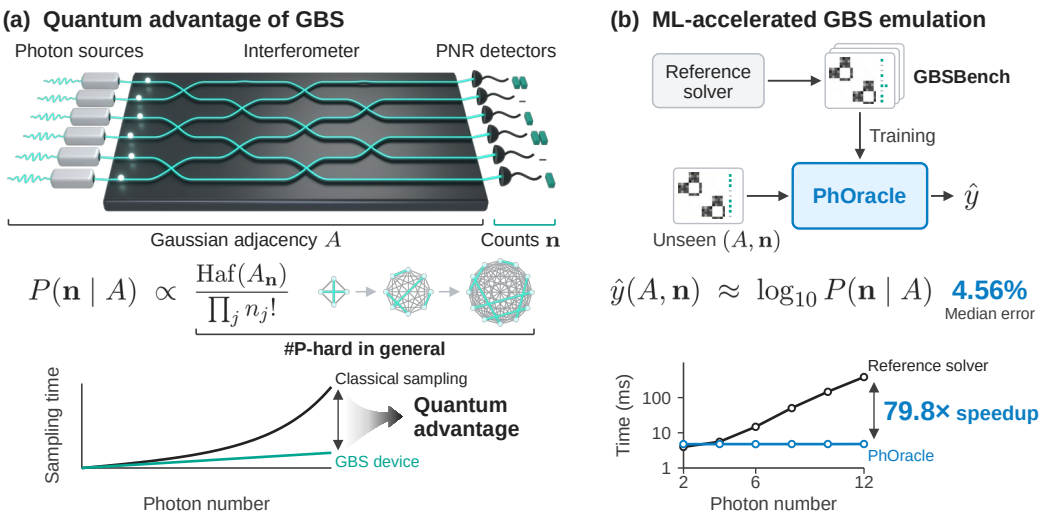

Figure 1: **Overview. (a)** GBS paves the way toward near-term quantum advantage. The Gaussian adjacency A represents the entire circuit, from the photon sources through the interferometer. GBS hardware directly samples photon-count patterns, whereas exact classical simulation involves costly Hafnian-based probability calculations. The graphs illustrate photon pairings, and the lower plot schematically contrasts classical and quantum sampling costs as photon number increases. **(b)** PhOracle accelerates GBS probability evaluation by learning from GBSBench to predict log-probabilities for unseen circuits and detection patterns. The lower plot compares measured probability-evaluation times for PhOracle and the reference solver.

Prior works have explored differentiable optimization of GBS distributions (Banchi et al., 2020b; Conti, 2021) and task-specific surrogates for photonic state preparation (Khanpour & Yeganeh, 2026). However, the extent to which learned probability estimators can generalize jointly across unseen circuits and arbitrary detection patterns remains largely unexplored. To address this gap, we consider a family of circuits governed by shared physical constraints and investigate whether their common structure can be exploited to learn output-probability evaluation across both circuits and detection patterns.

How accurately and efficiently can machine learning evaluate GBS probabilities, and how far does it generalize across circuits and detection patterns? We present a systematic study of learning GBS probabilities, from probability prediction to quantum circuit design. Our contributions are:

1. We formulate GBS probability evaluation as a learning task and present the *first systematic study* of its accuracy, computational cost, and generalization to unseen circuits and detection patterns.

2. We introduce **GBSBench**, the *first benchmark* for learned GBS probability evaluation, with reference data and baselines across a range of computational difficulties. Within a shared circuit family, its evaluation protocols test generalization to unseen interferometers, detection patterns, and photon-arrival conditions.

3. We propose **PhOracle**, a physics-informed recurrent model that adapts its computation to the requested detection pattern. It achieves a median absolute percentage error of 4.56% on unseen circuits and a speedup of up to 79.8× in probability evaluation over the reference solver once trained.

4. We show that the same PhOracle model, trained for probability prediction, can guide quantum circuit design without task-specific retraining. PhOracle provides a strong starting point for a short final refinement with the reference solver, accelerating single-pair and multipair Bell-state circuit discovery by 17.3× and 36.2×, respectively.

2 Related Work

Classical methods accelerate GBS simulation through more efficient sampling algorithms (Quesada & Arrazola, 2020; Quesada et al., 2022) and collision-aware calculations (Bulmer et al., 2022). Oh et al. (2024) used tensor networks to simulate lossy GBS experiments. More recently, Bulmer et al. (2026) developed algorithms for probabilities summed over unresolved internal modes. Their formulation underlies the grouped-PNR reference calculation in Section 3.

Machine learning acceleration techniques have been studied for lower-level hardware as well, such as individual nanophotonic devices and small photonic circuits. Learned surrogates have been developed to accelerate costly simulations in classical photonics, including neural solvers for Maxwell's equations such as NeurOLight (Gu et al., 2022) and PACE (Zhu et al., 2024). In quantum photonics, GLASE predicts the expected photon count at each output from interferometer phases for photonic classification (Notton et al., 2025).

GBS-related studies have also explored learning-based optimization: Banchi et al. (2020b) derived analytical gradients for tuning GBS distributions, while Conti (2021) used differentiable Gaussian calculations to optimize selected photon-detection events. Khanpour & Yeganeh (2026) trained a two-stage surrogate to select heralding patterns and predict fidelity and postselection probability for GKP-state preparation. PhOracle instead predicts the probability of a requested detection pattern from the circuit representation, allowing one evaluator to support different measurement queries and downstream objectives.

3 The Gaussian Boson Sampling Problem

In GBS, a Gaussian optical state defines a joint distribution over the photon counts recorded at all outputs (Hamilton et al., 2017). In this section, we specify the optical setting and state representation, then formulate the probability of a requested detection pattern as the learning target. Detailed mathematical derivations and physical formulations are provided in Appendix A.

Optical setting. We consider light generated by non-degenerate photon-pair sources, transformed by a linear-optical interferometer, and measured by photon-number-resolving (PNR) detectors (Figure 1(a)). Each detector records the total number of photons arriving at its output channel, which we call a *rail*. A measurement returns a detection pattern $\mathbf{n} = (n_1, \ldots, n_R)$ across R rails, with a total photon number $N = \sum_r n_r$. Several spectral modes can reach the same detector, which records their combined photon count without resolving their individual spectral profiles. We obtain the *coarse-grained PNR probability* by summing the probabilities of all photon assignments to unresolved internal modes that yield the observed rail counts (Bulmer et al., 2026).

State representation. We represent the zero-displacement Gaussian state before detection by the *Gaussian adjacency* $A \in \mathbb{C}^{2M\times 2M}$, a complex symmetric matrix of pair weights used to calculate photon-number probabilities (Hamilton et al., 2017; Kruse et al., 2019). Here $M = RK$ is the number of retained optical modes, with K internal modes per rail; the factor of two accounts for field coordinates and their conjugates. We order these doubled coordinates by rail throughout the probability and model formulas. The interferometer is represented by its unitary transfer matrix U, which can be physically implemented with beam splitters and phase shifters (Clements et al., 2016).

Learning target. A probability-generating function (Fitzke et al., 2023) encodes the photon-number distribution as a power series: the exponents identify each detection pattern, and the corresponding coefficient yields its probability. Introduce one auxiliary variable z_r per rail, collected in $\mathbf{z} = (z_1, \ldots, z_R)$. Each detected photon on rail r contributes one power of z_r. We define $E = XA$, where X is a fixed permutation matrix that swaps the first and last K rows within each rail's $2K$-row block. To attach the same variable to all doubled coordinates on a rail, we define the rail marker matrix $D(\mathbf{z})$ as:

$$D(\mathbf{z}) = \bigoplus_{r=1}^{R} z_r I_d, \qquad d = 2K, \tag{1}$$

where I_d is the $d \times d$ identity and $\bigoplus$ forms a block-diagonal matrix. Fully mode-resolved Gaussian probabilities are expressed through Hafnians, which sum products of pair weights over possible photon pairings. The Hafnian master theorem combines these probabilities into a determinant gen-

erating function $\mathcal{G}_A(\mathbf{z})$; assigning one variable to each rail collects all unresolved allocations with the same total count (Bulmer et al., 2026):

$$\mathcal{G}_A(\mathbf{z}) = \sum_{\mathbf{n}' \in \mathbb{N}_0^R} P(\mathbf{n}' \mid A) \prod_{r=1}^{R} z_r^{n_r'} = P_0 \det(I_{2M} - D(\mathbf{z})E)^{-1/2}. \tag{2}$$

Here the sum runs over all nonnegative integer detection patterns, and P_0 is the probability of detecting no photons on any rail. For example, the coefficient of $z_1 z_5$ gives the probability of one photon on rail 1, one on rail 5, and zero on every other rail. In general,

$$P(\mathbf{n} \mid A) = [z_1^{n_1} \cdots z_R^{n_R}]\, \mathcal{G}_A(\mathbf{z}), \tag{3}$$

where the brackets select the coefficient of the specified monomial. Our *reference solver* extracts this coefficient from a finite polynomial expansion of the coarse-grained generating function, avoiding a separate Hafnian evaluation for every internal-mode allocation (Bulmer et al., 2026). Given the Gaussian adjacency matrix A and a requested pattern $\mathbf{n}$, PhOracle predicts the log-probability $y = \log_{10} P(\mathbf{n}|A)$ to accommodate probabilities spanning many orders of magnitude.

4 PhOracle: A Physics-Informed Surrogate Model for GBS

PhOracle learns query-conditioned matrix compositions to predict log-probabilities. At each step, a shared Rail Transformer reads summaries of the current matrix state and the query to control rail weights and update magnitudes. The updated state returns to the Rail Transformer for the next composition. Recurrent matrix compositions build higher-order interactions, and a readout combines information across depths (Figure 2). Appendix B specifies the full architecture and training objective.

This design is motivated by coefficient extraction from a probability-generating function (Eqs. (2)–(3)). The reference calculation uses rail-marked matrix products, such as $ED(\mathbf{z})E$, and their traces through order N to obtain the probability of an N-photon pattern (Appendix A.6). PhOracle learns query-conditioned rail weights for analogous matrix compositions. We hypothesize that a small number of recurrent compositions can approximate this coefficient map across circuits and detection patterns.

Initialization. We initialize the matrix state $H^{(0)}$ from the rail-ordered matrix E. Its blocks $E_{rs} \in \mathbb{C}^{d\times d}$ map rail s to rail r, with $d = 2K$. Each block has four $K \times K$ sectors E_{rs}^{ab}, where a, b index the row and column halves, with 0 for annihilation and 1 for creation coordinates. We place the sectors of the normalized input $\widetilde{E}$ in separate channels to weight their interactions independently, and add an identity channel to preserve lower-order terms under multiplication:

$$H_{rs}^{(0)} = \left(\begin{bmatrix} \widetilde{E}_{rs}^{00} & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & \widetilde{E}_{rs}^{01} \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ \widetilde{E}_{rs}^{10} & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & \widetilde{E}_{rs}^{11} \end{bmatrix}, \delta_{rs} I_d \right). \tag{4}$$

Here δ_{rs} is the Kronecker delta. The $C = 5$ channels assemble into matrices $H_c^{(t)} \in \mathbb{C}^{Rd\times Rd}$ at depth t.

Rail Transformer. The Rail Transformer summarizes the evolving matrix state and the requested detection pattern to direct the next matrix composition. At depth t, traces, matrix inner products, and logarithmic scales contract the internal coordinates of $H^{(t)}$ into real rail features, which a shared MLP ϕ_H maps to state embeddings $\mathbf{e}_{H,r}^{(t)} \in \mathbb{R}^W$. The summaries are invariant under unitary changes of the unresolved internal-mode basis, matching the measurement symmetry, while the full complex matrices remain available for bilinear updates.

The query specifies both where photons are detected and the overall occupation structure, which we encode as local rail vectors $\mathbf{c}_r$ and a global vector $\mathbf{c}_{\mathrm{g}}$, respectively:

$$\begin{aligned} \mathbf{c}_r &= \left[n_r, \frac{n_r}{\max(N,1)}, \mathbf{1}[n_r > 0], \mathbf{1}[n_r > 1] \right], \\ \mathbf{c}_{\mathrm{g}} &= \left[N, \max_r n_r, \frac{1}{R}\sum_r \mathbf{1}[n_r > 0], \frac{1}{R}\sum_r \mathbf{1}[n_r > 1] \right]. \end{aligned} \tag{5}$$

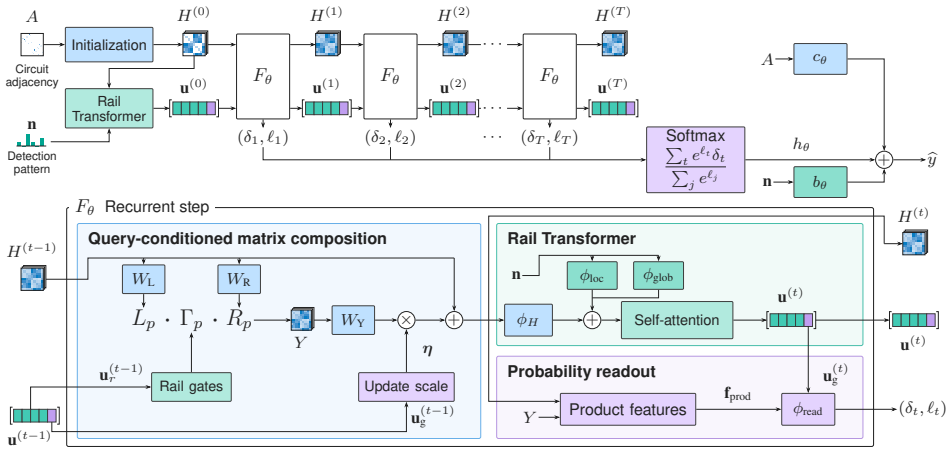

Figure 2: **PhOracle architecture.** *Top:* The Rail Transformer initializes $\mathbf{u}^{(0)}$ from $H^{(0)}$ and $\mathbf{n}$; a shared block F_θ then repeatedly updates H and $\mathbf{u}$. A softmax mixture of depth-wise corrections combines with circuit-only and query-only terms to predict the log-probability. *Bottom:* Controls guide the products $L_p\Gamma_p R_p$, stacked as Y, and the update to $H^{(t)}$. The Rail Transformer reads the updated state and fixed query embeddings to produce $\mathbf{u}^{(t)}$. The readout combines $\mathbf{u}_{\mathrm{g}}^{(t)}$ with product features $\mathbf{f}_{\mathrm{prod}}$ from $H^{(t)}$ and Y to predict a correction δ_t and mixing score ℓ_t.

Separate MLPs ϕ_{loc} and ϕ_{glob} map these vectors to width-W query embeddings, which remain fixed across recurrent depths. Each rail token combines $\mathbf{e}_{H,r}^{(t)}$ with $\phi_{\mathrm{loc}}(\mathbf{c}_r)$, while the global token combines the mean state embedding across rails with $\phi_{\mathrm{glob}}(\mathbf{c}_{\mathrm{g}})$. Shared self-attention layers process these $R+1$ tokens, returning rail controls $\mathbf{u}_r^{(t)} \in \mathbb{R}^W$ and a global control $\mathbf{u}_{\mathrm{g}}^{(t)} \in \mathbb{R}^W$. Self-attention allows each rail's gating signal to depend jointly on the complete detection pattern and the states of all other rails.

Query-Conditioned Matrix Composition. Each recurrent step advances the state from depth $t-1$ to t by composing two matrix factors through query-weighted intermediate rails. We first normalize the incoming state as $\overline{H}^{(t-1)} = H^{(t-1)}/\sigma_H^{(t-1)}$, where $\sigma_H^{(t-1)}$ is its root-mean-square (RMS) magnitude. Each recurrent step computes $J = 4$ matrix products in parallel, which are then mixed to update the state. Each branch $p = 1, \ldots, J$ forms its own left and right factors through learned complex channel maps W_{L} and W_{R}:

$$L_p^{(t-1)} = \sum_{c=1}^{C} (W_{\mathrm{L}})_{pc} \overline{H}_c^{(t-1)}, \qquad R_p^{(t-1)} = \sum_{c=1}^{C} (W_{\mathrm{R}})_{pc} \overline{H}_c^{(t-1)}. \tag{6}$$

The rail controls $\mathbf{u}_s^{(t-1)}$ determine real scalar gates $\gamma_{p,s}^{(t-1)}$ that weight each intermediate rail in the product $Y_p^{(t-1)} = L_p^{(t-1)} \Gamma_p^{(t-1)} R_p^{(t-1)}$, where $\Gamma_p^{(t-1)} = \bigoplus_s \gamma_{p,s}^{(t-1)} I_d$. This places the learned gates at the intermediate-rail positions occupied by the phase markers in the reference calculation:

$$\underbrace{\sum_{s=1}^{R} z_s E_{rs} E_{su}}_{(ED(\mathbf{z})E)_{ru}} \quad \longrightarrow \quad \underbrace{\sum_{s=1}^{R} \gamma_{p,s}^{(t-1)} (L_p^{(t-1)})_{rs} (R_p^{(t-1)})_{su}}_{(L_p^{(t-1)} \Gamma_p^{(t-1)} R_p^{(t-1)})_{ru}}. \tag{7}$$

Each term joins a relation from rail u to s with one from s to r. One branch fixes $\gamma_{1,s}^{(t-1)} = 1$ to keep unweighted compositions, while the remaining $J-1$ branches learn query-dependent rail weights.

The learned complex map W_{Y} combines the J product branches into the C state channels, forming a residual update:

$$\Delta H_c^{(t-1)} = \sum_{p=1}^{J} (W_{\mathrm{Y}})_{cp} Y_p^{(t-1)}. \tag{8}$$

The global control $\mathbf{u}_{\mathrm{g}}^{(t-1)}$ determines the update magnitudes and advances the state to depth t:

$$\boldsymbol{\eta}^{(t-1)} = \eta_{\max} \tanh(W_\eta \mathbf{u}_{\mathrm{g}}^{(t-1)} + \mathbf{b}_\eta),$$
$$H_c^{(t)} = H_c^{(t-1)} + \sigma_H^{(t-1)} \eta_c^{(t-1)} \frac{\Delta H_c^{(t-1)}}{\mathrm{RMS}(\Delta H_c^{(t-1)}) + \epsilon_{\mathrm{rms}}}. \tag{9}$$

The learned projection $(W_\eta, \mathbf{b}_\eta)$ yields channel-wise coefficients bounded in magnitude by $\eta_{\max}$, and $\epsilon_{\mathrm{rms}} > 0$ ensures numerical stability. Scaling by $\sigma_H^{(t-1)}$ keeps residual steps proportional to the incoming state scale.

We extract new rail features from $H^{(t)}$ and pass them through the same Rail Transformer with the fixed query embeddings. The resulting controls serve the current readout and the next matrix update. Each bilinear update can double the length of explicit matrix products, yielding paths with up to 2^T adjacency-derived factors after T recurrent steps. Sharing the Rail Transformer and composition parameters across these steps connects this design to looped Transformers (Giannou et al., 2023; Yang et al., 2024). The scalar action of Γ_p within each rail block commutes with rail-local changes of basis, preserving the internal-mode symmetry through every update.

Probability Readout. At each depth $t = 1, \ldots, T$, the readout predicts a scalar log-probability correction δ_t and mixing score ℓ_t. It combines the updated global control $\mathbf{u}_{\mathrm{g}}^{(t)}$ with invariant product features $\mathbf{f}_{\mathrm{prod}}^{(t)} \in \mathbb{R}^{2JC+J^2}$. These features contain complex inner products between the updated state $H^{(t)}$ and the products $Y_p^{(t-1)}$ that formed it, together with a Gram matrix of inner products among the product branches. A shared MLP ϕ_{read} evaluates their concatenation, and a softmax mixture combines the corrections:

$$(\delta_t, \ell_t) = \phi_{\mathrm{read}}\left([\mathbf{u}_{\mathrm{g}}^{(t)}; \mathbf{f}_{\mathrm{prod}}^{(t)}]\right),$$
$$h_\theta(A, \mathbf{n}) = \sum_{t=1}^{T} \frac{\exp(\ell_t)}{\sum_{j=1}^{T} \exp(\ell_j)} \delta_t. \tag{10}$$

Here θ denotes the trainable parameters.

The query head $b_\theta(\mathbf{n})$ uses pooled count embeddings, and the circuit head $c_\theta(A)$ uses the mean initial rail embedding. Their scalar predictions are added to the interaction term:

$$\widehat{y}(A, \mathbf{n}) = b_\theta(\mathbf{n}) + c_\theta(A) + h_\theta(A, \mathbf{n}). \tag{11}$$

We train PhOracle on reference log-probabilities and select checkpoints by validation Log-MSE.

5 GBSBench: Benchmarking GBS Probability Models

We introduce GBSBench to evaluate the accuracy, computational cost, and generalization of learned GBS probability models. It provides reference probabilities across varied optical settings and detection patterns, with queries selected to cover a range of computational difficulty. Appendix C provides construction and split details.

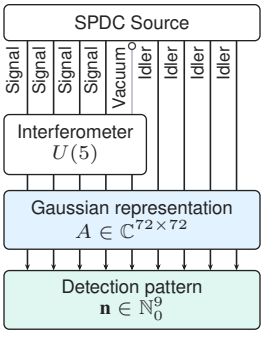

Figure 3: GBSBench dataset configuration.

5.1 Dataset Construction

Optical instances. GBSBench uses a nine-rail circuit family with four photon-pair sources and a programmable five-port interferometer (Figure 3). The interferometer's five output rails provide four logical rails for two dual-rail qubits and one ancillary rail for heralding, following previous Bell-state circuit designs (Fldzhyan et al., 2021; Hartnett et al., 2026). The sources feed four inputs of the interferometer, while the fifth input is vacuum. The four idler outputs bypass the interferometer, giving nine output rails in total. This configuration allows studying heralded entangled state generation and its application to circuit discovery in Section 7.

Table 1: GBSBench composition and temporal-mismatch evaluation. Test ID, pattern-OOD, and temporal mismatch share 800 interferometers.

Subset	Circuits	Photon total N	Queries per circuit	
			Per N	Total
Main composition				
Train	5,400	$2, 4, 6, 8, 10, 12$	16	96
Validation	800	$2, 4, 6, 8, 10, 12$	16	96
Test ID	800	$2, 4, 6, 8, 10, 12$	16	96
Test pattern-OOD	800	$8, 10, 12$	8	24
Additional OOD evaluation				
Temporal mismatch	800	$4, 6, 8, 10, 12$	Selected test queries	

Reference probabilities. Source energy and interferometer settings vary within this fixed layout. We model photon generation by spontaneous parametric down-conversion (SPDC) using spectrally resolved source propagation, thereby retaining multipair emission and spectral correlations in the learning target (Quesada et al., 2020; Kim et al., 2024). After linear-optical propagation, the reference solver evaluates detection probabilities using the formalism of Bulmer et al. (2026). For probability evaluation, each output rail is represented by four retained spectral modes. Changes in source properties and interference conditions are represented in the Gaussian adjacency. The dataset covers 25 source pumping energies from 0.001 to $20\,\mathrm{pJ}$, with equal numbers of circuits at each energy within a split. Five-port interferometers are sampled from the Haar distribution using the phase-corrected QR construction of Mezzadri (2007). For each circuit, queries are balanced across photon totals and sampled from occupation groups covering concentrated and spatially spread detection patterns.

5.2 Evaluation Protocols

Benchmark evaluation. Table 1 summarizes the circuit-disjoint training, validation, and test splits; each adjacency and its associated queries belong to a single split. On the 800 test circuits, in-distribution (ID) queries use the training pattern pools and test generalization to new interferometers. Pattern out-of-distribution (OOD) queries use count vectors excluded from training and validation.

All models use the same reference probabilities and validity masks, and checkpoints are selected by validation Log-MSE. Log-MSE measures squared error in base-10 log-probability; median absolute percentage error (Median APE) describes a typical relative error, while Mean APE captures the influence of large tail errors.

Temporal mismatch. Temporal mismatch evaluates whether PhOracle generalizes to changes in photon arrival times, beyond changes in the spatial interferometer. In practical photonic implementations, relative delays critically govern multi-source interference by altering photon temporal overlap and thus changing GBS output probabilities (Hong et al., 1987; Shi & Byrnes, 2022). In our spectrally resolved Gaussian model, a signal delay applies a frequency-dependent phase to the spectral modes, modifying the covariance and hence the adjacency matrix. Evaluation uses patterns from the original test set involving emission from at least two sources, where inter-source temporal overlap can affect interference.

6 Experiments

6.1 Experimental Setup

A query-only model excludes circuit information, and an additive circuit–query model excludes interactions between the two inputs. A residual MLP, query-conditioned CNN, and block Transformer jointly process the circuit and pattern. All comparisons use the splits and metrics defined in Section 5. PhOracle is trained with AdamW for 75,000 updates. Appendix D describes the baseline architectures, training procedures, and evaluation metrics. Appendix E presents additional experiments and analyses, including ablation studies.

Table 2: **Predictive performance on GBSBench test circuits.** Metrics aggregate the 96 ID and 24 pattern-OOD queries per circuit, using the common validity mask.

Method	Params (M)	Log-MSE ↓	Median APE (%) ↓	Mean APE (%) ↓
Query-only	0.884	16.61	99.62	5.21×10^{25}
Additive circuit–query	4.114	3.993	98.80	5.11×10^{13}
Residual MLP	7.736	0.9688	78.94	4.54×10^{9}
Query-conditioned CNN	5.730	0.1354	28.57	202.00
Block Transformer	5.971	0.1323	25.50	7.28×10^{5}
PhOracle	5.487	**0.01734**	**4.56**	**21.30**

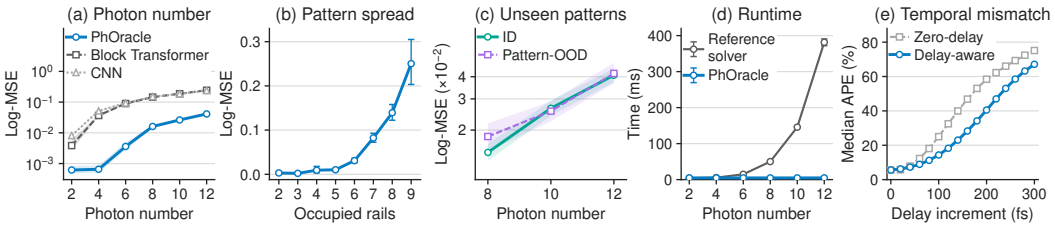

Figure 4: **Prediction accuracy and computational cost.** **(a)** Photon-number scaling for PhOracle and the two strongest baselines. **(b)** Twelve-photon ID queries grouped by occupied-rail count and **(c)** ID versus pattern-OOD queries at matched photon totals. **(d)** Mean wall time for calculating 16 queries over 800 validation circuits. **(e)** Zero-shot temporal mismatch with zero-delay and delay-aware inputs. Bands and bars in (a)–(d) show 95% circuit-bootstrap intervals.

6.2 Benchmark Results

Prediction accuracy. PhOracle achieves the lowest error on all three benchmark metrics on unseen test circuits (Table 2). The query-only and additive controls perform poorly, showing that neither pattern statistics alone nor an additive correction for the circuit captures the probability map. On unseen test interferometers, PhOracle remains more accurate than the CNN and block Transformer throughout the two- to twelve-photon range (Figure 4(a)). At twelve photons, errors increase when photons occupy more rails (Figure 4(b)), contributing to the gap between mean and median APE. At matched photon totals, PhOracle achieves similar errors on ID and held-out detection patterns with familiar collision signatures (Figure 4(c)), indicating generalization to new photon allocations.

Computational cost. PhOracle's evaluation time remains nearly constant as photon number increases, while the reference solver becomes more expensive (Figure 4(d)). PhOracle is faster from four photons onward, reaching a $79.8\times$ speedup at twelve photons on the same GPU.

Temporal mismatch. Updating the adjacency to include the delay reduces median APE without retraining. Figure 4(e) compares the same frozen model with its input held at zero delay or updated to include the delay, both against the delayed reference calculated by the reference solver. The model thus responds to delay-induced changes in interference through the spectral information encoded in its input. Appendix E.3 describes the delay setting.

Ablations. Reducing recurrence to one update, even with a deeper Rail Transformer, raises Log-MSE by $5.20\times$; restricting query conditioning to the readout raises it by $7.25\times$. These results support using the detection pattern to guide successive matrix compositions, with benefits that are not recovered by a deeper single-step Rail Transformer or readout-only conditioning.

7 Quantum Photonic Circuit Discovery

Bell states provide basic entanglement resources for photonic quantum information processing, but designing circuits to generate them requires repeated, costly evaluations of multiphoton detection probabilities. We use PhOracle to accelerate this search by predicting the probabilities needed to optimize Bell-state fidelity, without task-specific retraining. Extending the circuit-search method of Hartnett et al. (2026), we consider both single-pair and multipair generation cases. The single-

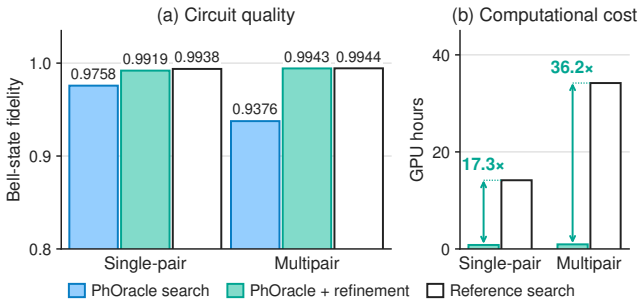

Figure 5: **Bell-state circuit discovery.** **(a)** Fidelities of the circuits found by PhOracle, before and after refinement, compared with reference-solver search. **(b)** Accumulated GPU hours for 75 searches per method and task.

pair task involves eight detected photons across idler and interferometer outputs; the multipair task involves ten and requires a two-photon idler herald (Tiedau et al., 2019).

Search and Refinement. Quantum-state tomography reconstructs a density matrix from measurement probabilities collected across different analyzer settings (Banchi et al., 2018). Following Hartnett et al. (2026), we minimize the Bures angle to maximize Bell-state fidelity and parameterize the interferometer unitary through its Lie algebra. Gradients propagate through the density-matrix reconstruction to the 24 real-valued unitary parameters. During PhOracle-assisted search, these probabilities are supplied by the frozen PhOracle model; the reference-solver comparison uses the same reconstruction and objective. Appendix F details the state reconstruction and verification procedure.

For each task, both methods search all 25 source energies with three matched random initializations per energy and a common convergence rule. PhOracle's score selects a single candidate from the 75 searches, followed by a short reference-solver refinement. Circuit-discovery times reported here cover all searches and the final refinement. Appendix G specifies the search and timing protocols.

Results. PhOracle identifies initial circuits with rescored fidelities of 0.9758 for the single-pair task and 0.9376 for the multipair task. After a brief reference-solver refinement, the fidelities reach 0.9919 and 0.9943, on par with full reference searches that achieve 0.9938 and 0.9944, as shown in Figure 5(a). All reported fidelities are evaluated using reference-solver probabilities, and the reconstructed states pass positivity and measurement-consistency checks both before and after refinement. At comparable final fidelities, PhOracle-assisted search achieves end-to-end speedups of $17.3\times$ on the single-pair task and $36.2\times$ on the multipair task (Figure 5(b)).

8 Conclusion

We presented the first systematic study of learning GBS probabilities, using PhOracle and GBSBench to evaluate accuracy, computational cost, and generalization. A single trained PhOracle model generalizes to unseen circuits and detection patterns and guides Bell-state circuit discovery without task-specific retraining.

Limitations. PhOracle and GBSBench are not structurally restricted to the rail and photon-number ranges considered here: the same formulation can in principle be applied to datasets with larger interferometers and higher photon numbers. However, our work intentionally focuses on a controlled circuit family to establish the benchmark and baseline behavior. How accurately and efficiently learned probability models generalize to larger systems and new physical settings remains open.

Outlook. Extending the benchmark to realistic sources and larger interferometers would test learned probability evaluation across a broader range of spectral and spatial complexity. Such extensions could enable searches currently limited by simulation cost, including source–interferometer co-design. Moreover, since Hafnian-based computations also arise in fields such as molecular vibronic spectroscopy (Quesada, 2019) and molecular docking (Banchi et al., 2020a), our model design can motivate developing learned approximations for a broader class of repeated combinatorial calculations.

AI USE STATEMENT

Generative AI tools assisted with literature review, refinement of research hypotheses and mathematical formulations, feedback on benchmark and experimental design, method implementation and debugging, dataset-generation and processing pipelines, experiment execution, and interpretation of results. They also assisted with figure preparation and manuscript drafting, translation, and editing. Reference probability labels were computed by the numerical solver. AI assistance concerned the implementation and operation of the data pipeline. AI-assisted code and calculations were checked through implementation tests, numerical cross-checks, and comparison with the cited literature. The authors reviewed the AI-assisted contributions and take responsibility for the final methods, results, and manuscript.

REPRODUCIBILITY STATEMENT

Appendices A–G specify the physical model, benchmark construction, model configuration, optimization protocol and timing methodology. Code, configurations, and instructions for training, evaluation, and circuit discovery will be made available anonymously during peer review.

REFERENCES

Alán Aspuru-Guzik, Anthony D Dutoi, Peter J Love, and Martin Head-Gordon. Simulated quantum computation of molecular energies. *Science*, 309(5741):1704–1707, 2005.

Leonardo Banchi, W Steven Kolthammer, and M. S. Kim. Multiphoton tomography with linear optics and photon counting. *Physical Review Letters*, 121(25):250402, 2018.

Leonardo Banchi, Mark Fingerhuth, Tomas Babej, Christopher Ing, and Juan Miguel Arrazola. Molecular docking with gaussian boson sampling. *Science Advances*, 6(23):eaax1950, 2020a.

Leonardo Banchi, Nicolás Quesada, and Juan Miguel Arrazola. Training gaussian boson sampling distributions. *Physical Review A*, 102(1):012417, 2020b.

Jacob F. F. Bulmer, Bryn A Bell, Rachel S Chadwick, Alex E Jones, Diana Moise, Alessandro Rigazzi, Jan Thorbecke, Utz-Uwe Haus, Thomas Van Vaerenbergh, Raj B Patel, et al. The boundary for quantum advantage in gaussian boson sampling. *Science Advances*, 8(4):eabl9236, 2022.

Jacob F. F. Bulmer, Javier Martínez-Cifuentes, Bryn A Bell, and Nicolás Quesada. Simulating lossy and partially distinguishable quantum optical circuits: theory, algorithms, and applications to experiment validation and state preparation. *Advanced Photonics*, 8(1):016010, 2026.

Earl T Campbell, Barbara M Terhal, and Christophe Vuillot. Roads towards fault-tolerant universal quantum computation. *Nature*, 549(7671):172–179, 2017.

William R Clements, Peter C Humphreys, Benjamin J Metcalf, W Steven Kolthammer, and Ian A Walmsley. Optimal design for universal multiport interferometers. *Optica*, 3(12):1460–1465, 2016.

Claudio Conti. Training gaussian boson sampling by quantum machine learning. *Quantum Machine Intelligence*, 3(2):26, 2021.

Pim de Haan, Maurice Weiler, Taco Cohen, and Max Welling. Gauge equivariant mesh CNNs: Anisotropic convolutions on geometric graphs. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Jnspzp-oIZE.

Tom Dodd, Javier Martínez-Cifuentes, Oliver Thomson Brown, Nicolás Quesada, and Raúl García-Patrón. A fast and frugal gaussian boson sampling emulator. *arXiv preprint arXiv:2511.14923*, 2025.

Erik Fitzke, Florian Niederschuh, and Thomas Walther. Simulating the photon statistics of multimode gaussian states by automatic differentiation of generating functions. *APL Photonics*, 8(2): 026106, 02 2023. ISSN 2378-0967. doi: 10.1063/5.0129638. URL https://doi.org/10.1063/5.0129638.

Suren A Fldzhyan, Mikhail Yu Saygin, and Sergei P Kulik. Compact linear optical scheme for Bell state generation. *Physical Review Research*, 3(4):043031, 2021.

Angeliki Giannou, Shashank Rajput, Jy-Yong Sohn, Kangwook Lee, Jason D Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *International Conference on Machine Learning*, pp. 11398–11442. PMLR, 2023.

Jiaqi Gu, Zhengqi Gao, Chenghao Feng, Hanqing Zhu, Ray Chen, Duane Boning, and David Pan. NeurOLight: A physics-agnostic neural operator enabling parametric photonic device simulation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 14623–14636. Curran Associates, Inc., 2022. doi: 10.52202/068431-1063. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/5ddfb189c022a317ff1c72e6639079de-Paper-Conference.pdf.

Brajesh Gupt, Josh Izaac, and Nicolás Quesada. The Walrus: a library for the calculation of hafnians, Hermite polynomials and gaussian boson sampling. *Journal of Open Source Software*, 4(44): 1705, 2019.

Craig S. Hamilton, Regina Kruse, Linda Sansoni, Sonja Barkhofen, Christine Silberhorn, and Igor Jex. Gaussian boson sampling. *Phys. Rev. Lett.*, 119:170501, Oct 2017.

Gavin S Hartnett, Dave Kielpinski, Smarak Maity, Pranav S Mundada, Yuval Baum, and Michael R Hush. Automated discovery of heralded ballistic graph state generators for fusion-based photonic quantum computation. *Physical Review A*, 113(4):042608, 2026.

C. K. Hong, Z. Y. Ou, and L. Mandel. Measurement of subpicosecond time intervals between two photons by interference. *Phys. Rev. Lett.*, 59:2044–2046, Nov 1987. doi: 10.1103/PhysRevLett.59.2044. URL https://link.aps.org/doi/10.1103/PhysRevLett.59.2044.

Mohammad Amin Khanpour and Hossein Davoodi Yeganeh. Rapid gaussian boson sampling circuit screening for GKP states creation via a two-stage machine learning surrogate, 2026. URL https://arxiv.org/abs/2606.05992.

Seonghun Kim, Youngbin Kim, Young-Do Yoon, Seongjin Jeon, Woo-Joo Kim, and Young-Ik Sohn. Simulation of integrated nonlinear quantum optics: From nonlinear interferometer to temporal walk-off compensator. *Phys. Rev. Appl.*, 22:064092, 2024.

Regina Kruse, Craig S Hamilton, Linda Sansoni, Sonja Barkhofen, Christine Silberhorn, and Igor Jex. Detailed study of gaussian boson sampling. *Physical Review A*, 100(3):032326, 2019.

Lars S Madsen, Fabian Laudenbach, Mohsen Falamarzi Askarani, Fabien Rortais, Trevor Vincent, Jacob F. F. Bulmer, Filippo M Miatto, Leonhard Neuhaus, Lukas G Helt, Matthew J Collins, et al. Quantum computational advantage with a programmable photonic processor. *Nature*, 606(7912): 75–81, 2022.

Francesco Mezzadri. How to generate random matrices from the classical compact groups. *Notices of the American Mathematical Society*, 54(5):592 – 604, May 2007. ISSN 0002-9920.

Cassandre Notton, Vassilis Apostolou, Agathe Senellart, Anthony Walsh, Daphne Wang, Yichen Xie, Songqinghao Yang, Ilyass Mejdoub, Oussama Zouhry, Kuan-Cheng Chen, et al. Establishing baselines for photonic quantum machine learning: Insights from an open, collaborative initiative. *arXiv preprint arXiv:2510.25839*, 2025.

Changhun Oh. Recent theoretical and experimental progress on boson sampling. *Current Optics and Photonics*, 9(1):1–18, 2025.

Changhun Oh, Minzhao Liu, Yuri Alexeev, Bill Fefferman, and Liang Jiang. Classical algorithm for simulating experimental gaussian boson sampling. *Nature Physics*, 20(9):1461–1468, 2024.

Nicolás Quesada and Juan Miguel Arrazola. Exact simulation of gaussian boson sampling in polynomial space and exponential time. *Physical Review Research*, 2(2):023005, 2020.

Nicolás Quesada and J. E. Sipe. Effects of time ordering in quantum nonlinear optics. *Physical Review A*, 90(6):063840, 2014.

Nicolás Quesada, Gil Triginer, Mihai D Vidrighin, and J. E. Sipe. Theory of high-gain twin-beam generation in waveguides: From Maxwell's equations to efficient simulation. *Physical Review A*, 102(3):033519, 2020.

Nicolás Quesada, Rachel S. Chadwick, Bryn A. Bell, Juan Miguel Arrazola, Trevor Vincent, Haoyu Qi, and Raúl García−Patrón. Quadratic speed-up for simulating gaussian boson sampling. *PRX Quantum*, 3:010306, Jan 2022. doi: 10.1103/PRXQuantum.3.010306. URL https://link.aps.org/doi/10.1103/PRXQuantum.3.010306.

Nicolás Quesada. Franck-Condon factors by counting perfect matchings of graphs with loops. *The Journal of Chemical Physics*, 150(16):164113, 04 2019. ISSN 0021-9606. doi: 10.1063/1.5086387. URL https://doi.org/10.1063/1.5086387.

Raffaele Santagati, Alan Aspuru-Guzik, Ryan Babbush, Matthias Degroote, Leticia González, Elica Kyoseva, Nikolaj Moll, Markus Oppel, Robert M Parrish, Nicholas C Rubin, et al. Drug design on quantum computers. *Nature Physics*, 20(4):549–557, 2024.

Junheng Shi and Tim Byrnes. Effect of partial distinguishability on quantum supremacy in gaussian boson sampling. *npj Quantum Information*, 8(1):54, 2022.

Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):303–332, 1999.

Johannes Tiedau, Tim J Bartley, Georg Harder, Adriana E Lita, Sae Woo Nam, Thomas Gerrits, and Christine Silberhorn. Scalability of parametric down-conversion for generating higher-order Fock states. *Physical Review A*, 100(4):041802, 2019.

Christian Weedbrook, Stefano Pirandola, Raúl García-Patrón, Nicolas J Cerf, Timothy C Ralph, Jeffrey H Shapiro, and Seth Lloyd. Gaussian quantum information. *Reviews of Modern Physics*, 84(2):621–669, 2012.

Liu Yang, Kangwook Lee, Robert Nowak, and Dimitris Papailiopoulos. Looped transformers are better at learning learning algorithms. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 42195–42214, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/b8402301e7f06bdc97a31bfaa653dc32-Paper-Conference.pdf.

Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, et al. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020.

Hanqing Zhu, Wenyan Cong, Guojin Chen, Shupeng Ning, Ray T Chen, Jiaqi Gu, and David Z Pan. PACE: Pacing operator learning to accurate optical field simulation for complicated photonic devices. *Advances in Neural Information Processing Systems*, 37:67535–67555, 2024.

A PHYSICAL FORMALISM

We use a spectrally resolved Gaussian simulation pipeline to generate reference probabilities and formulate learning in terms of the resulting optical state. This provides a common representation for circuits with different source and component properties. Changes in source spectra, relative delays, frequency-dependent transmission, and fabrication-induced shifts can be incorporated through the output covariance while retaining the same photon-count probability prediction task. Our source calculation includes finite-gain multipair emission and time-ordered propagation (Quesada et al., 2020; Kim et al., 2024). The underlying Gaussian framework also supports pump self-phase modulation and pump-induced cross-phase modulation, providing a basis for extending GBSBench to more realistic device conditions (Kim et al., 2024).

We first describe the source output and its propagation through the interferometer using Gaussian covariance matrices. We then reduce the spectral representation on each detector rail, convert the retained covariance into a Gaussian adjacency matrix, and use it to calculate photon-count probabilities.

A.1 GAUSSIAN CONVENTIONS

A mode specifies a spatial and spectral field profile. Several spectral modes can reach the same rail detector. These spectral modes provide a discrete representation of a continuous spectrum for photons with finite bandwidth. A Gaussian state is specified by the means and covariances of two field quadratures per optical mode, the quantum counterparts of the real and imaginary components of a complex optical amplitude (Weedbrook et al., 2012). We use an equivalent representation in terms of annihilation and creation operators. The annihilation operator $\hat{a}_j$ removes one photon from mode j, while the creation operator $\hat{a}_j^\dagger$ adds one. For M_{full} modes before spectral reduction, we collect both operators from each mode in the column

$$\hat{\boldsymbol{\xi}} = \left(\hat{a}_1, \ldots, \hat{a}_{M_{\mathrm{full}}}, \hat{a}_1^\dagger, \ldots, \hat{a}_{M_{\mathrm{full}}}^\dagger\right)^\top. \tag{12}$$

We use $\dagger$ for conjugate transpose, $\top$ for transpose, and $*$ for elementwise conjugation. Our sources have zero displacement, meaning that the mean optical field is zero, so their Gaussian states are fully specified by the covariance

$$V_{jk} = \frac{1}{2}\left\langle \Delta\hat{\xi}_j \Delta\hat{\xi}_k^\dagger + \Delta\hat{\xi}_k^\dagger \Delta\hat{\xi}_j \right\rangle, \qquad V_{\mathrm{vac}} = \frac{1}{2} I_{2M_{\mathrm{full}}}. \tag{13}$$

Here $\langle\cdot\rangle$ denotes an expectation in the optical state, $\Delta\hat{\xi}_j = \hat{\xi}_j - \langle\hat{\xi}_j\rangle$, and j, k index the $2M_{\mathrm{full}}$ doubled coordinates.

A.2 SPDC SOURCE PROPAGATION

We use the frequency-domain high-gain SPDC model of Quesada et al. (2020); Kim et al. (2024), with an undepleted classical pump, vacuum signal and idler inputs, and lossless propagation without self- or cross-phase modulation. In matched spectral bases for the signal and idler arms, the source output is a product of two-mode squeezed-vacuum states. Each arm has $N_\omega = 401$ frequency bins. The numerical implementation composes transfer matrices over 500 propagation cells, retaining the poling sign, accumulated phase mismatch, and the order of interactions. Each cell combines free propagation with a nonlinear exponential containing first-order pair generation and a second-order dispersion–coupling correction. The fixed poling pattern follows the Gaussian accumulated-nonlinearity target of Kim et al. (2024).

In SPDC, the output annihilation operators depend on both input annihilation and input creation operators. The creation-operator contribution describes photon-pair generation. We denote the coefficient matrices of the annihilation and creation contributions by F_{src} and G_{src}, respectively, with src denoting the source. Both matrices have size $2N_\omega \times 2N_\omega$, covering the signal and idler frequency bins. For source energy ε, the corresponding doubled transformation and vacuum-output covariance are

$$S_{\mathrm{src}}^{(\varepsilon)} = \begin{pmatrix} F_{\mathrm{src}} & G_{\mathrm{src}} \\ G_{\mathrm{src}}^* & F_{\mathrm{src}}^* \end{pmatrix}, \qquad V_{\mathrm{src}}^{(\varepsilon)} = \frac{1}{2} S_{\mathrm{src}}^{(\varepsilon)} S_{\mathrm{src}}^{(\varepsilon)\dagger}. \tag{14}$$

This covariance retains source brightness, spectral correlations, and multipair emission without imposing a fixed photon-number input.

A.3 NINE-RAIL CIRCUIT

The input covariance $V_{\mathrm{in}}^{(\varepsilon)}$ combines four independent copies of $V_{\mathrm{src}}^{(\varepsilon)}$ and one vacuum rail, with no correlations between sources. We place all annihilation coordinates before all creation coordinates, using rail order $[s_0, s_1, s_2, s_3, i_0, i_1, i_2, i_3, v]$ in each half. Here s_q, i_q label the two arms of source $q = 0, \ldots, 3$, and v is the vacuum rail. Rail counts use one-based indices: rails 1–4 are the signal ports, rails 5–8 are the idlers, and rail 9 is the additional port with a vacuum input. A five-port interferometer U mixes the four signal rails and the vacuum rail, while the four idler rails bypass it unchanged. The same unitary is applied independently at each frequency bin, with no mixing between frequencies. We denote the full transformation in the doubled representation by S_U, which applies this transformation to the annihilation coordinates and its complex conjugate to the creation coordinates. The output covariance is

$$V_{\mathrm{circ}}^{(\varepsilon,U)} = S_U V_{\mathrm{in}}^{(\varepsilon)} S_U^\dagger, \tag{15}$$

with side $2 \times 9 \times 401 = 7218$ before detector reduction.

Pair creation and passive propagation conserve the signal–idler number difference. The supported rail-count queries therefore satisfy

$$\sum_{r\in\{1,2,3,4,9\}} n_r = \sum_{r\in\{5,6,7,8\}} n_r = \frac{N}{2}, \qquad N = \sum_{r=1}^{9} n_r \in 2\mathbb{N}_0, \tag{16}$$

where n_r is the count on rail r, N is the total, and $\mathbb{N}_0$ denotes the nonnegative integers.

A.4 RAIL-LOCAL REDUCTION

The 401-bin grid resolves source propagation but gives a large covariance matrix for repeated photon-count probability calculations. For the sources used here, pair correlations are concentrated in a few spectral modes. We therefore construct a spectral basis independently on each of the $R = 9$ output rails after interference, then retain its dominant modes (Figure 6). Each detector measures the total photon number in its rail, which is invariant under unitary changes of spectral basis within that rail.

Let $(\mathsf{C}_{rs})_{ij} = \langle \hat{a}_{r,i} \hat{a}_{s,j} \rangle$ be the pair correlation between spectral coordinates i, j on rails r, s. We diagonalize the positive-semidefinite Gram matrix

$$G_r^{\mathrm{det}} = \sum_{s=1}^{R} \mathsf{C}_{rs} \mathsf{C}_{rs}^\dagger = W_r \Lambda_r W_r^\dagger, \qquad \Lambda_r = \mathrm{diag}(\lambda_{r,1}, \ldots, \lambda_{r,N_\omega}). \tag{17}$$

The columns of W_r are orthonormal eigenvectors, ordered by decreasing eigenvalue $\lambda_{r,k}$. We fix the number of retained modes at $K_r = 4$ on every rail, selecting the eigenvectors with the largest pair-correlation weights. This compact multimode representation includes weaker spectral contributions alongside the dominant mode and has the same dimension for every circuit.

Let W_{red} be the block-diagonal matrix of these retained eigenvectors, and define its doubled form $S_{\mathrm{red}} = W_{\mathrm{red}} \oplus W_{\mathrm{red}}^*$. The subscript red denotes the reduced representation containing only the retained spectral modes. The retained covariance is

$$V_{\mathrm{red}} = S_{\mathrm{red}}^\dagger V_{\mathrm{circ}} S_{\mathrm{red}}, \qquad M_{\mathrm{red}} = \sum_{r=1}^{R} K_r. \tag{18}$$

Tracing out the discarded spectral modes preserves the Gaussian form of the retained state. Counting only the retained modes, however, approximates the full rail-count distribution.

To assess this spectral cutoff, we compute the retained pair-correlation weight as the sum of the four leading Gram eigenvalues divided by the sum of all eigenvalues. At the highest source energy of 20 pJ, this fraction is 99.9824% in both source arms.

A.5 GAUSSIAN ADJACENCY

The Gaussian adjacency represents the output state in a form used to calculate photon-count probabilities. Following the Gaussian photon-counting convention of Hamilton et al. (2017); Kruse et al.

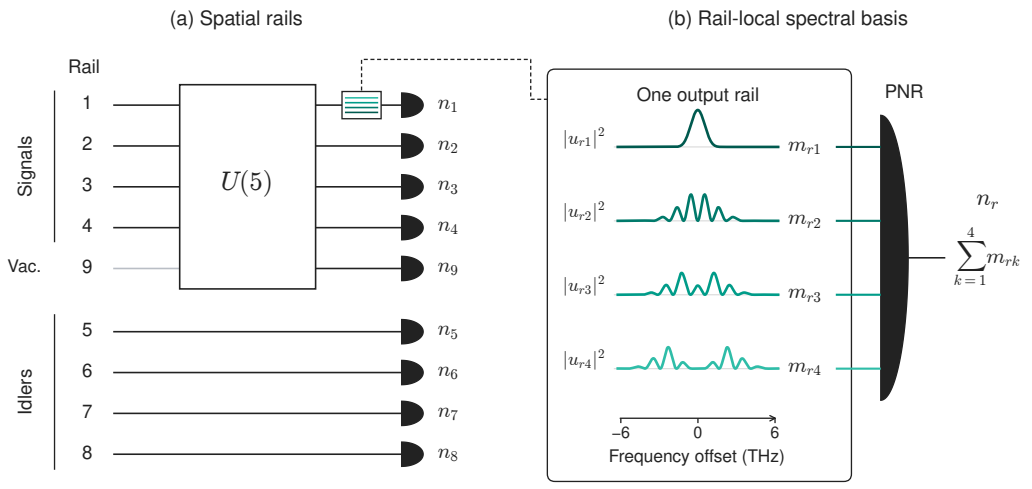

Figure 6: **Spatial rails and unresolved spectral modes.** Four signal inputs and one vacuum input pass through the interferometer, and the four idlers bypass it. The enlarged output rail contains four retained spectral basis functions, each spanning multiple frequency bins. The profiles show the squared magnitudes of the four retained spectral modes on output rail 1 of a validation circuit at 5 pJ. Each mode u_{rk} corresponds to column k of W_r defined below, with its plotted squared magnitude normalized to unit area in frequency. All profiles share a common intensity scale. A PNR detector records only the total $n_r = \sum_{k=1}^{4} m_{rk}$, where m_{rk} is the occupation of spectral mode k in rail r. The colored lines in the enlarged rail are its spectral modes.

(2019), define

$$Q_{\mathrm{red}} = V_{\mathrm{red}} + \frac{1}{2} I_{2M_{\mathrm{red}}}, \qquad X_{\mathrm{red}} = \begin{pmatrix} 0 & I_{M_{\mathrm{red}}} \\ I_{M_{\mathrm{red}}} & 0 \end{pmatrix}. \tag{19}$$

The shifted covariance Q_{red} and coordinate-exchange matrix X_{red} give

$$A_{\mathrm{red}} = X_{\mathrm{red}} \left(I_{2M_{\mathrm{red}}} - Q_{\mathrm{red}}^{-1} \right), \qquad A_{\mathrm{red}} = A_{\mathrm{red}}^{\top}. \tag{20}$$

The joint vacuum probability is

$$P_0 = \frac{1}{\sqrt{\det Q_{\mathrm{red}}}} = \sqrt{\det(I_{2M_{\mathrm{red}}} - X_{\mathrm{red}} A_{\mathrm{red}})}, \tag{21}$$

using the positive square root.

With $K = 4$ modes per rail, $M_{\mathrm{red}} = M = RK = 36$, and the stored matrix is $A_{\mathrm{store}} = A_{\mathrm{red}} \in \mathbb{C}^{72\times 72}$. The stored order places all annihilation coordinates before their creation counterparts (Figure 7). The implementation first symmetrizes the stored adjacency by averaging it with its transpose to remove finite-precision asymmetry. In the coordinate convention used for the probability and model formulas, a real permutation Π groups each rail's K annihilation and K creation coordinates:

$$A = \Pi A_{\mathrm{store}} \Pi^{\top}, \qquad X = \bigoplus_{r=1}^{R} \begin{pmatrix} 0 & I_K \\ I_K & 0 \end{pmatrix}. \tag{22}$$

In the resulting rail order, $E = XA$ is the matrix used throughout the probability and model formulas.

Displaced Gaussian states can be incorporated by augmenting the state representation with the mean optical field and including the corresponding displacement terms in the probability calculation (Bulmer et al., 2026).

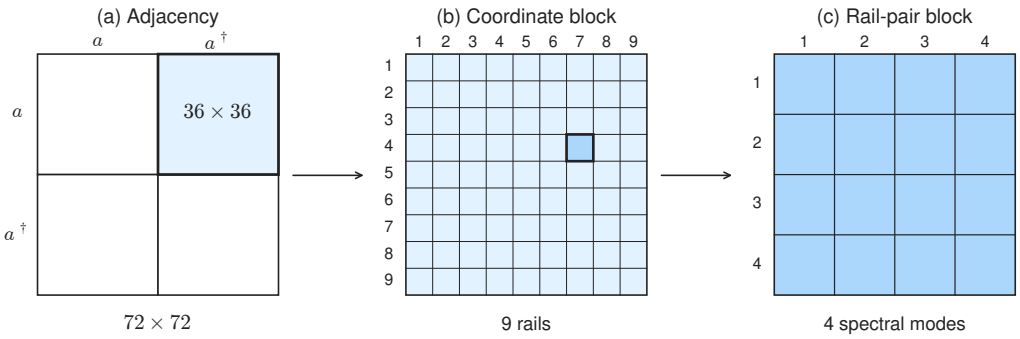

Figure 7: **Stored adjacency and its block structure.** The coordinate sectors are shown before the model's rail-wise permutation.

A.6 Grouped PNR Probabilities

For a symmetric matrix $B \in \mathbb{C}^{2N\times 2N}$, the Hafnian is

$$\mathrm{Haf}(B) = \sum_{\pi\in \mathrm{PM}(2N)} \prod_{(i,j)\in\pi} B_{ij}, \tag{23}$$

where $\mathrm{PM}(2N)$ contains the perfect pairings of all $2N$ indices. For a mode-resolved occupation $\mathbf{m} \in \mathbb{N}_0^M$, repeat both conjugate coordinates of mode j in A exactly m_j times to obtain $A_{\mathbf{m}} \in \mathbb{C}^{2N\times 2N}$, with $N = \sum_j m_j$. Repeated indices account for collisions, or multiple photons in the same mode. The zero-mean Gaussian probability is (Hamilton et al., 2017; Kruse et al., 2019)

$$p(\mathbf{m} \mid A) = P_0 \frac{\mathrm{Haf}(A_{\mathbf{m}})}{\prod_{j=1}^M m_j!}. \tag{24}$$

Summing unresolved spectral modes. Let $\mathcal{I}_r$ contain the spectral modes on rail r. The detector reports their total n_r, giving the grouped probability

$$P(\mathbf{n} \mid A) = \sum_{\mathbf{m}:\sum_{j\in\mathcal{I}_r} m_j = n_r \ \forall r} p(\mathbf{m} \mid A). \tag{25}$$

Lowercase p denotes a fully mode-resolved probability, while uppercase P denotes its rail-grouped marginal. Unresolved occupations are mutually exclusive outcomes, so their probabilities add.

The zero-displacement specialization of the loop-Hafnian master theorem collects this sum into the determinant generating function in equation 2 (Bulmer et al., 2026). Assigning the same marker z_r to every mode on rail r combines all allocations with total n_r into the coefficient of $z_r^{n_r}$. Our kernel extracts that coefficient without enumerating the individual Hafnians.

Trace powers and the finite photon cutoff. We use the rail-marker matrix $D(\mathbf{z})$ defined in equation 1. An auxiliary variable κ marks total photon number, and the determinant has the formal expansion (Bulmer et al., 2026)

$$\Phi_A(\kappa, \mathbf{z}) = \det(I_{2M} - \kappa D(\mathbf{z})E)^{-1/2} \tag{26}$$

$$= \exp\left[\sum_{k=1}^{\infty} \frac{\kappa^k}{2k} \mathrm{Tr}\big([D(\mathbf{z})E]^k\big)\right]. \tag{27}$$

Here Φ_A omits the vacuum factor P_0. Its coefficient of κ^N contains all patterns with total photon number N. Terms with $k > N$ cannot contribute, so only

$$g_k(\mathbf{z}) = \frac{1}{2k} \mathrm{Tr}\big([D(\mathbf{z})E]^k\big), \quad 1 \le k \le N, \qquad a_N(\mathbf{z}) = [\kappa^N]\Phi_A(\kappa, \mathbf{z}) \tag{28}$$

are needed. For the benchmark's largest photon total, the kernel therefore stops at $(DE)^{12}$ without truncating any term that contributes to the requested probability.

For $j = 0, \ldots, N$ and $k = 1, \ldots, N$, let $a_j^{(k)}$ be the coefficient of κ^j after including exponent terms through $g_k \kappa^k$. Multiplying each factor $\exp(g_k \kappa^k)$ into the coefficient vector gives the implemented recurrence:

$$a_j^{(k)}(\mathbf{z}) = \sum_{u=0}^{\lfloor j/k \rfloor} a_{j-ku}^{(k-1)}(\mathbf{z}) \frac{g_k(\mathbf{z})^u}{u!}, \qquad a_0^{(0)}(\mathbf{z}) = 1, \qquad a_N(\mathbf{z}) = a_N^{(N)}(\mathbf{z}). \tag{29}$$

All other initial coefficients are zero, and only degrees $0, \ldots, N$ are retained. The index u counts copies of the order-k term, leaving degree $j - ku$ for the previous coefficient vector.

We then extract the individual rail counts from the homogeneous degree-N polynomial a_N using a roots-of-unity filter. For rail r, set $s_r = n_r + 1$ and $\omega_r = \exp(2\pi i / s_r)$. The weighted Fourier sum is

$$P(\mathbf{n} \mid A) = \frac{P_0}{\prod_{r=1}^{R} s_r} \sum_{j_1=0}^{s_1-1} \cdots \sum_{j_R=0}^{s_R-1} \left(\prod_{r=1}^{R} \omega_r^{-j_r n_r} \right) a_N\left(\omega_1^{j_1}, \ldots, \omega_R^{j_R}\right). \tag{30}$$

The index j_r selects a root of unity, and the phase weights retain only exponents congruent to n_r modulo $n_r + 1$. Any other surviving nonnegative exponent would exceed n_r. Since a_N has total degree $N = \sum_r n_r$, only the requested pattern remains.

For a query evaluated separately, the root grid contains $\prod_r (n_r + 1)$ points, each requiring matrix powers through order N.

Numerical cross-check. The reference solver supplies both the labels used to train and evaluate PhOracle and the probabilities used to verify discovered circuits. To check its numerical implementation, we used The Walrus, an open-source Python library for Hafnian calculations and Gaussian boson sampling (Gupt et al., 2019). We compared 76,800 reference probabilities from the 800 validation circuits against those computed by its grouped-PNR implementation on the `internal_modes` development branch. Both implementations used the same detector-reduced Gaussian states and detection patterns, so the comparison isolates the probability-evaluation step. The maximum absolute difference was 4.37×10^{-16}. For reference probabilities above 10^{-12}, the maximum relative difference was 2.58×10^{-6}. This close agreement supports the numerical reliability of our grouped-PNR implementation.

B Model Details

We specify the input dimensions, rail-pair representation, and state and query initialization, followed by the recurrent update, readout, and training objective.

B.1 Dimensions and Inputs

Let R denote the number of detector rails and K the number of retained internal modes on each rail. The physical mode count is $M = RK$, and the doubled width of one rail is $d = 2K$. We use C for the number of matrix-state channels, J for the number of parallel product branches, and T for recurrent depth. Each input consists of the complex $2M \times 2M$ rail-ordered adjacency A defined in (22) and an R-entry detection pattern $\mathbf{n}$. The model returns one real-valued log-probability per pattern and can evaluate multiple patterns for the same adjacency.

B.2 Doubled Coordinates and Rail-Pair Matrix Blocks

In rail ordering, the block E_{rs} maps the doubled coordinate space $\mathcal{D}_s \cong \mathbb{C}^d$ to $\mathcal{D}_r$. Left multiplication by X exchanges the annihilation and creation row sectors within each rail block, giving $E = XA$. The model uses $\widetilde{E} = E/s_A$, where s_A is the fixed training-set adjacency RMS. Matrix multiplication sums over the intermediate rail and its internal coordinates.

The measured rail total is invariant to changes of internal-mode basis. For $U_r \in U(K)$, define

$$\mathcal{U}_r = \mathrm{diag}(U_r, \overline{U}_r). \tag{31}$$

The rail blocks transform as

$$\widetilde{E}_{rs} \longmapsto \mathcal{U}_r \widetilde{E}_{rs} \mathcal{U}_s^\dagger. \tag{32}$$

Intermediate basis transformations cancel in matrix products, and closed-product traces are invariant.

B.3 Initial Matrix State

Let P_0 and P_1 select the first and second K coordinates within one doubled rail:

$$\mathsf{P}_0 = \begin{bmatrix} I_K & 0 \\ 0 & 0 \end{bmatrix}, \qquad \mathsf{P}_1 = \begin{bmatrix} 0 & 0 \\ 0 & I_K \end{bmatrix}. \tag{33}$$

The four data channels in Eq. (4) are the projections $S_{rs}^{ab} = \mathsf{P}_a \widetilde{E}_{rs} \mathsf{P}_b$, for $a, b \in \{0, 1\}$. These channels sum to $\widetilde{E}$ but can be weighted independently. The identity channel preserves lower-order terms in bilinear products. The projectors enforce matching intermediate sectors:

$$S_{rs}^{ab} S_{su}^{cd} = \delta_{bc}\, \mathsf{P}_a \widetilde{E}_{rs} \mathsf{P}_b \widetilde{E}_{su} \mathsf{P}_d. \tag{34}$$

The circuit's initial state is computed once and broadcast over its queries.

B.4 Rail Features and Query Tokens

Invariant contractions summarize the complex matrix state as real rail tokens for the Rail Transformer. The full matrices remain available for subsequent products.

At every depth, the state is divided by a global RMS:

$$\sigma_H^{(t)} = \left[\frac{1}{CR^2 d^2} \sum_{c,r,s} \|H_{c,rs}^{(t)}\|_F^2 + \epsilon_{\mathrm{rms}} \right]^{1/2}, \qquad \overline{H}^{(t)} = H^{(t)} / \sigma_H^{(t)}. \tag{35}$$

For channel c, define

$$\begin{aligned} \sigma_{H,c}^{(t)} &= \left[\frac{1}{R^2 d^2} \sum_{r,s} \|H_{c,rs}^{(t)}\|_F^2 + \epsilon_{\mathrm{rms}} \right]^{1/2}, \\ \mathbf{s}_H^{(t)} &= \left[\log(\sigma_H^{(t)} + \epsilon_{\mathrm{rms}}), \left\{ \log(\sigma_{H,c}^{(t)} + \epsilon_{\mathrm{rms}}) - \log(\sigma_H^{(t)} + \epsilon_{\mathrm{rms}}) \right\}_{c=1}^{C} \right]. \end{aligned} \tag{36}$$

Here $\epsilon_{\mathrm{rms}} = 10^{-8}$ is the fixed stabilizer used by all RMS normalizations. The vector $\mathbf{s}_H^{(t)} \in \mathbb{R}^{C+1}$ retains the global state scale and the relative channel scales. For each rail, we also compute

$$\begin{aligned}
\tau_{r,c}^{(t)} &= \frac{1}{d}\,\mathrm{Tr}\Big(\overline{H}_{c,rr}^{(t)}\Big), \\
G_{r,cc'}^{\mathrm{out},(t)} &= \frac{1}{Rd^2}\sum_s \mathrm{Tr}\Big(\overline{H}_{c,rs}^{(t)}\overline{H}_{c',rs}^{(t)\dagger}\Big), \\
G_{r,cc'}^{\mathrm{in},(t)} &= \frac{1}{Rd^2}\sum_s \mathrm{Tr}\Big(\overline{H}_{c,sr}^{(t)\dagger}\overline{H}_{c',sr}^{(t)}\Big), \\
C_{r,cc'}^{\mathrm{cyc},(t)} &= \frac{1}{Rd^2}\sum_s \mathrm{Tr}\Big(\overline{H}_{c,rs}^{(t)}\overline{H}_{c',sr}^{(t)}\Big).
\end{aligned} \tag{37}$$

The vector $\boldsymbol{\tau}_r^{(t)} = (\tau_{r,c}^{(t)})_{c=1}^{C}$ records self-block traces. G_r^{out} and G_r^{in} compare channels sharing an output or input rail, respectively. The oriented contraction C_r^{cyc} retains the complex trace of each outward–return product. For either Hermitian Gram matrix, hvec packs the real diagonal and complex upper triangle into C^2 real coordinates. We concatenate the real and imaginary trace components, packed Gram matrices, real and imaginary cyclic-product components, and scale features into $\mathbf{f}_r^{(t)} \in \mathbb{R}^{4C^2+3C+1}$. These contractions are unchanged by the rail-local coordinate transformation in Eq. (32).

Separate MLPs ϕ_{loc} and ϕ_{glob} map the local and global vectors in Eq. (5) to the common Rail Transformer width W. Both query embeddings are computed once and reused at every depth and in the query-only prediction head. The state projection maps this feature vector to the Rail Transformer width: $\mathbf{e}_{H,r}^{(t)} = \phi_H(\mathbf{f}_r^{(t)}) \in \mathbb{R}^W$.

The R rail tokens and one global token at depth t are

$$\begin{aligned}
\mathbf{x}_r^{(t)} &= \mathbf{e}_{H,r}^{(t)} + \phi_{\mathrm{loc}}(\mathbf{c}_r) + \mathbf{e}_{\mathrm{rail}} + \mathbf{e}_t, \\
\mathbf{x}_{\mathrm{g}}^{(t)} &= \frac{1}{R}\sum_{r=1}^{R}\mathbf{e}_{H,r}^{(t)} + \phi_{\mathrm{glob}}(\mathbf{c}_{\mathrm{g}}) + \mathbf{e}_{\mathrm{global}} + \mathbf{e}_t.
\end{aligned} \tag{38}$$

The learned vectors $\mathbf{e}_{\mathrm{rail}}$ and $\mathbf{e}_{\mathrm{global}}$ distinguish rail tokens from the global token. Every rail uses the same $\mathbf{e}_{\mathrm{rail}}$. The learned vector $\mathbf{e}_t$ identifies the current recurrent depth and is added to every token at that depth. Self-attention then lets every rail condition its control signal on the complete detection pattern and the current state of the other rails. The shared pre-normalized Rail Transformer returns a width-W global output $\mathbf{u}_{\mathrm{g}}^{(t)}$ and one width-W rail output $\mathbf{u}_r^{(t)}$ for each rail. No rail-index embedding is added. A joint permutation of the rail blocks and count vector therefore permutes the rail outputs and leaves the global output unchanged.

B.5 Query-Conditioned Matrix Composition

The block combines state-derived left and right factors with rail-token gates and global-token residual coefficients.

The coefficient matrices $W_{\mathrm{L}}, W_{\mathrm{R}}$ in Eq. (6) each contain $J \times C$ complex parameters. Each resulting factor has $R \times R$ blocks of size $d \times d$. Combining its rail and internal-mode indices gives a $2M \times 2M$ matrix. The implementation includes one unweighted product. The remaining rail gates are predicted from the rail outputs of the Rail Transformer:

$$\begin{aligned}
&\gamma_{1,r}^{(t-1)} = 1, \qquad \gamma_{p,r}^{(t-1)} = \tanh\Big((\mathbf{w}_p^{\Gamma})^{\top}\mathbf{u}_r^{(t-1)} + b_p^{\Gamma}\Big), \quad p = 2,\dots,J, \\
&\Gamma_p^{(t-1)} = \bigoplus_{r=1}^{R}\gamma_{p,r}^{(t-1)} I_d.
\end{aligned} \tag{39}$$

Here $\mathbf{w}_p^{\Gamma} \in \mathbb{R}^W$ and $b_p^{\Gamma} \in \mathbb{R}$ are the learned weight vector and bias for branch p, shared across rails and recurrent steps. In Eq. (7), the gate scales the right factor's rows before multiplication by the left factor. The complex mixing matrix W_{Y} has shape $C \times J$, and $W_\eta \in \mathbb{R}^{C\times W}$ produces one

residual coefficient per channel in Eq. (9). The RMS in that update averages over the R^2d^2 entries of each update channel, with the same stabilizer as in Eq. (35). All update parameters are shared across depths. The depth embedding $\mathbf{e}_t$ distinguishes the stages, and each stage acts on the updated matrix state.

Composition depth. The trace expansion in Eq. (27) can be written in rail blocks as

$$\mathrm{Tr}\big((D(\mathbf{z})E)^k\big) = \sum_{r_1,\ldots,r_k} \left(\prod_{j=1}^{k} z_{r_j}\right) \mathrm{Tr}(E_{r_1r_2}E_{r_2r_3}\cdots E_{r_kr_1})\,. \tag{40}$$

Each monomial records the rail visits in a closed matrix product. The requested coefficient combines such products, motivating recurrent matrix composition with query-dependent intermediate-rail weights.

Counting adjacency-derived matrix factors while treating scalar gates and normalization factors as coefficients, each bilinear update can double the maximum product length. Starting from one factor, five updates allow products of up to 32 factors. Residual addition and the identity channel retain shorter products.

Coordinate symmetry. Gauge-equivariant networks enforce consistent transformations under changes of local coordinate frames (de Haan et al., 2021). The recurrent core preserves the rail-local coordinate transformation. Since Γ_p is scalar within each rail, it commutes with $\mathcal{U} = \bigoplus_r \mathcal{U}_r$, and

$$L'_p\Gamma_pR'_p = \mathcal{U}(L_p\Gamma_pR_p)\mathcal{U}^\dagger. \tag{41}$$

The hidden-state matrices transform covariantly, and the contractions supplied to scalar prediction heads are invariant.

B.6 Post-Update Readout

The Rail Transformer is evaluated on the initial state and after every update, giving six evaluations for $T = 5$. At depth t, the readout combines the products $Y^{(t-1)}$ with the Rail Transformer output from the updated state $H^{(t)}$. This output also controls the next update.

First, all J product matrices share one RMS scale,

$$\sigma_Y^{(t-1)} = \left[\frac{1}{J(2M)^2}\sum_{p=1}^{J}\|Y_p^{(t-1)}\|_F^2 + \epsilon_{\mathrm{rms}}\right]^{1/2}, \qquad \overline{Y}_p^{(t-1)} = Y_p^{(t-1)}/\sigma_Y^{(t-1)}. \tag{42}$$

One common scale preserves the relative magnitudes of the product branches. With the normalized updated state $\overline{H}^{(t)}$, the readout computes

$$\begin{aligned} C_{pc}^{\mathrm{YH},(t)} &= \frac{1}{(2M)^2}\,\mathrm{Tr}\Big(\overline{Y}_p^{(t-1)}\overline{H}_c^{(t)\dagger}\Big)\,, \\ G_{pp'}^{\mathrm{prod},(t)} &= \frac{1}{(2M)^2}\,\mathrm{Tr}\Big(\overline{Y}_p^{(t-1)}\overline{Y}_{p'}^{(t-1)\dagger}\Big)\,. \end{aligned} \tag{43}$$

C^{YH} compares new products with updated channels, and G^{prod} compares product branches. The real and imaginary entries of C^{YH} and the Hermitian packing of G^{prod} provide $2JC + J^2 = 56$ real features. Concatenating them with the width-320 global token gives the 376-dimensional input to a shared MLP, which predicts a correction δ_t and a depth logit ℓ_t for each $t = 1,\ldots,T$. The initial state is used to produce the first update but is not itself read out. Equation (10) applies softmax over the T depth scores and sums the corresponding predictions.

Additive prediction heads.

The query-only head $b_\theta(\mathbf{n})$ concatenates the global query embedding with the mean local query embedding, then applies an affine map, layer normalization, and an MLP. The circuit-only head $c_\theta(A)$ applies a separate MLP to the mean of the initial rail-feature embeddings. These scalar outputs are added to the interaction prediction as in Eq. (11).

B.7 Objective and Model Configuration

For each valid circuit–query pair i, the residual is $e_i = \widehat{y}_i - y_i$, where y_i is the reference log-probability in base ten. Queries sharing a circuit and total photon number form a group $\mathcal{I}_g$, with mean residual $\overline{e}_g = |\mathcal{I}_g|^{-1} \sum_{i \in \mathcal{I}_g} e_i$ and $n_{\mathrm{valid}} = \sum_g |\mathcal{I}_g|$ valid pairs in the minibatch. We augment log-MSE with a within-group spacing term:

$$\begin{aligned} \mathcal{L}_{\mathrm{value}} &= \frac{1}{n_{\mathrm{valid}}} \sum_i e_i^2, \\ \mathcal{L}_{\mathrm{spacing}} &= \frac{1}{n_{\mathrm{valid}}} \sum_g \sum_{i \in \mathcal{I}_g} (e_i - \overline{e}_g)^2, \\ \mathcal{L} &= \mathcal{L}_{\mathrm{value}} + \lambda_{\mathrm{spacing}} \mathcal{L}_{\mathrm{spacing}}. \end{aligned} \tag{44}$$

The auxiliary term penalizes within-group residual variance. We use $\lambda_{\mathrm{spacing}} = 0.10$ and select checkpoints by validation Log-MSE alone.

Table 3: PhOracle model configuration.

Quantity	Value
Rails R / modes per rail K / doubled width d	$9/4/8$
Maximum training photon number	12
State channels C / product branches J / recurrent steps T	$5/4/5$
Rail Transformer width / heads / layers / FFN width	$320/8/3/1280$
Rail-feature projection	$116 \to 640 \to 320$
Local and global query projections	$4 \to 320 \to 320$
Depth readout	$376 \to 768 \to 768 \to 2$
Dropout	0.05
Maximum / initial residual update ratio	$0.20/0.03$
Trainable parameters	$5{,}486{,}916$

The network outputs an unconstrained real value in base-10 log space. Downstream probabilities are obtained as $10^{\widehat{y}}$.

C GBSBENCH DETAILS

We describe the source settings, Haar-random interferometers, query sampling, and circuit-disjoint splits.

C.1 SOURCE AND CIRCUIT

Each circuit uses four identical copies of one source covariance, with four idlers bypassing the five-port signal interferometer. The stored output order is $[u_0, u_1, u_2, u_3, i_0, i_1, i_2, i_3, u_4]$, with u labeling interferometer outputs and i labeling bypassed idlers. The 25 source energies are 0.001, 0.01, 0.1, 0.3, 0.5, and 1–20 pJ in integer increments. Sub-pJ points resolve the low-gain regime more densely. The upper range was chosen from the mean photon numbers to cover useful source brightness. The source was designed for high spectral purity at low gain. Its geometry, poling pattern, pump bandwidth, and chirp are fixed across the energy sweep. The numerical source design uses a waveguide width of $1.1848\,\mu\mathrm{m}$, a length of $5\,\mathrm{mm}$, and a nominal apodized quasi-phase-matching period of $2.4202\,\mu\mathrm{m}$. Pump bandwidth and chirp are fixed at $8.5 \times 10^{12}\,\mathrm{rad\,s^{-1}}$ and zero, respectively. The effective SPDC coupling coefficient is fixed at $100\,\mathrm{W^{-1/2}\,m^{-1}}$ across all source energies. It sets the strength of pump-driven signal–idler pair generation, while the gain is varied through the pump pulse energy.

At each energy, we compute the full source covariance using time-ordered high-gain propagation (Figure 8) (Quesada & Sipe, 2014; Quesada et al., 2020). Spectral reduction is applied after propagation through the interferometer.

Source statistics. We compute source brightness and spectral concentration (Table 4) from the singular values s_j of the full signal–idler pair-creation transfer kernel. For vacuum input, define

$$\bar{n}_{\mathrm{s}} = \sum_j s_j^2, \qquad w_j = \frac{s_j^2}{\bar{n}_{\mathrm{s}}}, \qquad \mathcal{P}_{\mathrm{src}} = \sum_j w_j^2, \qquad K_{\mathrm{src}} = \mathcal{P}_{\mathrm{src}}^{-1}. \tag{45}$$

Here $\bar{n}_{\mathrm{s}}$ is the mean signal photon number per pulse from one source. The spectral purity $\mathcal{P}_{\mathrm{src}}$ measures the concentration of the normalized first-order coherence spectrum, and K_{src} is the effective Schmidt number, distinct from the integer detector-retention dimension K_r. All four source copies have the same statistics at a given energy.

C.2 HAAR INTERFEROMETERS

For each circuit seed, we draw $X, Y \in \mathbb{R}^{5\times 5}$ independently with standard-normal entries and form $Z = (X + iY)/\sqrt{2}$. If $Z = QR$ is a complex QR factorization, we set

$$\Lambda = \mathrm{diag}\left(\frac{R_{11}}{|R_{11}|}, \ldots, \frac{R_{55}}{|R_{55}|}\right), \qquad U = Q\Lambda. \tag{46}$$

The diagonal phase correction yields the Haar distribution (Mezzadri, 2007). Each unitary and its split-unique seed are stored with the circuit.

C.3 QUERY BANK

Each training and validation circuit receives 16 queries at each total photon number $N \in \{2, 4, 6, 8, 10, 12\}$, for 96 queries per circuit. Test circuits receive the same allocation of ID queries and eight additional pattern-OOD queries at each of $N = 8, 10, 12$, for 120 queries per circuit.

The query pools are frozen before circuit-level sampling. For each photon total, supported count vectors are grouped by collision signature, defined by the sorted nonzero occupations on the signal and idler sides separately. The training pool includes every supported collision signature. At $N = 8, 10, 12$, count vectors within each signature are divided into disjoint ID and pattern-OOD pools. For each circuit and photon total, a seeded sampler cycles through signature groups from a random starting offset and draws a pattern within each visited group. Duplicate vectors within a circuit's stratum are replaced with unused candidates. Supported queries obey Eq. (16), expressed in the dataset's zero-based rail convention.

Table 4: **Source statistics across the 25 operating points.** Values are computed from the full source transfer kernel before detector-mode reduction. Mean photon number is per signal arm, per pulse, for one source.

Energy (pJ)	Spectral purity $\mathcal{P}_{\mathrm{src}}$	Effective Schmidt number K_{src}	Mean photon number $\bar{n}_{\mathrm{s}}$
0.001	0.994570	1.005459	4.1205×10^{-5}
0.01	0.994572	1.005457	4.1212×10^{-4}
0.1	0.994594	1.005435	0.0041288
0.3	0.994641	1.005388	0.012437
0.5	0.994685	1.005343	0.020814
1	0.994788	1.005240	0.042060
2	0.994954	1.005071	0.085879
3	0.995073	1.004951	0.13153
4	0.995147	1.004877	0.17910
5	0.995178	1.004846	0.22865
6	0.995169	1.004855	0.28027
7	0.995123	1.004901	0.33405
8	0.995042	1.004982	0.39008
9	0.994930	1.005096	0.44843
10	0.994788	1.005240	0.50922
11	0.994618	1.005411	0.57252
12	0.994424	1.005608	0.63845
13	0.994206	1.005828	0.70711
14	0.993968	1.006068	0.77860
15	0.993712	1.006328	0.85303
16	0.993438	1.006605	0.93052
17	0.993150	1.006897	1.0112
18	0.992849	1.007202	1.0951
19	0.992537	1.007519	1.1825
20	0.992215	1.007846	1.2735

Within a signature, count vectors differ by permutations of occupations within the signal and idler sides. For a fixed circuit, these permutations describe different detection events and generally have different probabilities. This differs from consistently relabeling both the circuit representation and the count vector, which leaves the physical event and its probability unchanged.

C.4 Circuit-Disjoint Splits

Before numerical filtering, the training, validation, and test splits contain 518,400, 76,800, and 96,000 circuit–query pairs, respectively, for 691,200 in total. The common validity mask excludes 26 pairs with nonpositive reference probabilities, for which the log-probability is undefined. The test split comprises 76,800 ID and 19,200 pattern-OOD pairs. Every source energy contributes 216 training, 32 validation, and 32 test circuits. Haar seeds are disjoint across all splits, and every query attached to one adjacency follows that adjacency.

Input normalization is fitted on training data, and validation selects the recipe and checkpoint. The circuit-disjoint test split is excluded from training and selection.

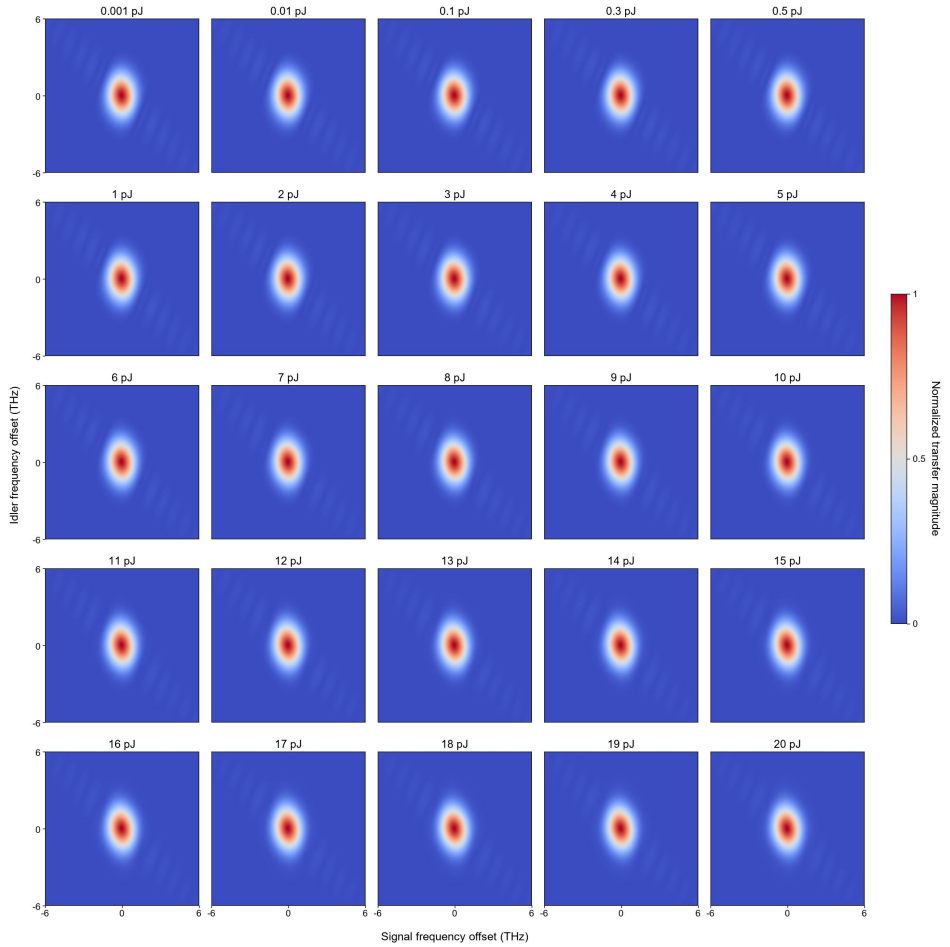

Figure 8: **Source transfer functions across the 25 operating points.** Signal–idler pair-creation kernel magnitudes at the 25 source energies, each normalized by its own maximum.

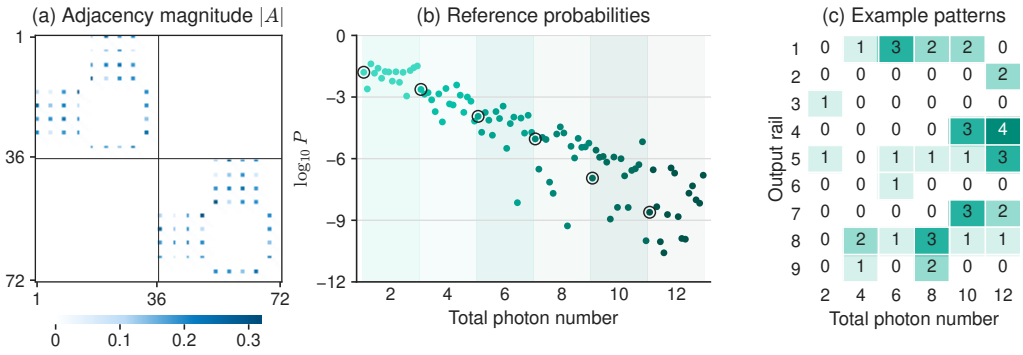

Figure 9: **One GBSBench circuit and its probability queries.** **(a)** Adjacency magnitude, **(b)** 96 reference probabilities with 16 queries per photon total, and **(c)** example detection patterns corresponding to the outlined points in (b). The validation circuit was selected independently of model predictions. Probabilities are not renormalized.

D EXPERIMENTAL SETUP

We describe the baseline models, training procedure, and evaluation metrics.

D.1 BASELINE MODELS

We choose baselines to test the need for circuit–pattern interactions and compare general-purpose architectures for learning them (Table 5). The query-only control measures how much can be predicted from count-pattern statistics without circuit information. The additive circuit–query control tests whether independent circuit and pattern contributions suffice. The residual MLP, query-conditioned CNN, and block Transformer compare dense regression on the full adjacency, convolutional processing of the rail-pair grid, and query-to-circuit attention, respectively. All baselines share PhOracle's inputs, validity masks, and splits, except that the query-only control omits the adjacency.

Table 5: **Baseline architectures and trainable parameter counts.**

Model	Core configuration	Parameters
Query-only	Query width 384; MLP hidden width 640	883,521
Additive circuit–query	Block width 64; circuit hidden width 512	4,113,538
Residual MLP	State width 400; three residual blocks, hidden width 800	7,735,878
Query-conditioned CNN	Block projection widths 48/96/192/256; two residual convolution blocks	5,729,555
Block Transformer	Width 320; eight heads; two encoder and two cross-attention layers; FFN width 1,280	5,970,851

Shared input encoding. The ordered count vector is embedded rail by rail using 64-dimensional count embeddings and four scalar features: normalized count, fraction of total photons, occupancy, and collision indicators. A shared MLP produces 64-dimensional rail features, which are combined with learned rail positions and four global count summaries. The resulting query representation has width 384, except in the block Transformer, where it has width 320. Circuit inputs retain all 10,368 real and imaginary adjacency entries, either flattened or reshaped into 81 rail-pair blocks of 128 scalars. The models directly predict log-probability without a fixed photon-total intercept derived from training labels.

Query-only. The query encoder and a scalar MLP head predict from the detection pattern alone.

Additive circuit–query. The query branch matches the query-only control. A shared $128 \to 256 \to 64$ block encoder processes each rail pair, and a circuit network combines the concatenated block features. The circuit and query branches produce scalar outputs that are added.

Residual MLP. The flattened adjacency is projected to a hidden state, with a separate projection of its log root-mean-square magnitude to retain overall scale. The query is added to the initial state and modulates each residual block through learned feature-wise scaling and shifting. The interaction readout uses a hidden width of 512.

Query-conditioned CNN. A shared block projection produces a 9×9 feature grid, augmented with learned row and column positions. Residual blocks apply 3×3 convolutions with group normalization on this grid. Row- and column-specific query features and a global query embedding condition both the pooling weights and the features being pooled. A width-960 fusion network with a width-384 bottleneck predicts the interaction term.

Block Transformer. Each rail-pair block is embedded with learned row and column positions and a separate projection of its log root-mean-square magnitude. A global token additionally receives the overall adjacency scale. Encoder layers process these 82 tokens, and cross-attention layers use the query embedding to read the encoded circuit. The query encoder has a hidden width of 608.

D.2 Training and Implementation

All neural models train for 75,000 updates using AdamW with a global batch size of 32 circuits, weight decay 10^{-4}, and gradient clipping at 1.0. AdamW uses momentum coefficients $(0.9, 0.999)$ and a numerical stabilizer of 10^{-8}. A 3,750-step linear warmup is followed by cosine decay to 0.02 of the peak learning rate. Training uses bfloat16 automatic mixed precision, and the model dropout probability is 0.05. PhOracle uses a peak learning rate of 4.5×10^{-4} and a spacing-loss weight of 0.10. The baselines keep their fixed training recipes with a peak learning rate of 6×10^{-4} and a spacing-loss weight of 0.25. All models select checkpoints by validation Log-MSE, evaluated every 500 updates. The PhOracle training configuration uses eight devices with four circuits per device and a query chunk size of 96. Training the selected checkpoint took 3.90 hours on eight RTX 4090 GPUs (31.20 GPU hours). The adjacency scale is computed from the training split and fixed for validation and test evaluation. Minibatches retain complete query groups for each sampled circuit so that Eq. (44) is evaluated on its intended groups. The released configurations specify the remaining implementation settings.

D.3 Evaluation and Reporting

We report Log-MSE, the mean squared error in base-10 log-probability, together with median and mean APE over all valid queries. An absolute log error δ corresponds to a multiplicative probability factor of 10^{δ}. For predicted log-probability $\hat{y}$ and reference y, absolute percentage error (APE) is $100|10^{\hat{y}} - 10^{y}|/10^{y}$ on valid labels. Paired confidence intervals resample circuits rather than treating query rows as independent replicates.

E Additional Results and Analysis

We report prediction and scaling diagnostics, runtime measurements, temporal-mismatch results, and component ablations.

E.1 Prediction and Scaling

The occupied-rail groups in Figure 4(b) contain 135–3,213 queries. Circuit bootstrapping retains all queries from each resampled circuit. Matching photon totals in the ID/pattern-OOD comparison avoids differences caused by the pooled query families' photon-number mixtures.

E.2 Probability-Evaluation Runtime

We compare PhOracle and the reference solver on one NVIDIA RTX PRO 6000 Blackwell Server Edition GPU, hosted by an Intel Xeon 6952P CPU. We reuse the fixed checkpoint and all 800 validation circuits, with sixteen queries at each of six photon totals, giving 76,800 probability evaluations per implementation. Each implementation is timed once per bank after warmup, with sequential execution in cyclic order. The timed intervals include query-plan or input preparation, transfers, synchronized computation, and output transfer, but exclude source simulation, adjacency construction, model loading, and JIT compilation. PhOracle inference uses CUDA Graph replay with fresh inputs for each evaluation. All 76,800 predictions are bitwise identical to those obtained in eager execution with unchanged model weights. The one-time graph setup takes 0.777 seconds and is excluded from the per-bank timings. Including it changes the aggregate speedup over all banks from $21.0\times$ to $20.3\times$. Each speedup is the ratio of mean reference-solver time to mean PhOracle time across paired circuits. The 95% confidence intervals use 2,000 complete-circuit bootstrap replicates and describe variation across circuits. All queries are retained in the timing measurements without clipping or filtering. These measurements concern forward probability evaluation. Circuit-discovery timings are measured separately and do not use this graph replay.

E.3 Temporal Mismatch

Delay model. We reuse all 800 test interferometers and 25 source energies, keeping the trained PhOracle checkpoint fixed. In our spectrally resolved Gaussian model, a signal delay applies a frequency-dependent phase to the spectral modes, modifying the covariance and hence the adjacency matrix. Before the interferometer, the four signal paths receive delays $0, \Delta t, 2\Delta t, 3\Delta t$, with $\Delta t = 0, 20, \ldots, 300\,\mathrm{fs}$. The idler paths are unchanged. Thus the largest pairwise delay is $900\,\mathrm{fs}$. A $10\,\mathrm{pJ}$ pilot guided this range. The inverse RMS angular-frequency bandwidth of the leading signal mode was approximately $309\,\mathrm{fs}$, providing a characteristic scale for changes in temporal overlap. We use the same absolute delay grid at all source energies. We propagate the full source covariance through the delays and interferometer, then recompute the rail-local detector bases with four retained modes per rail.

By contrast, a strictly single-frequency squeezer model without an explicit wavepacket or distinguishability parameter cannot represent delay-induced loss of temporal overlap because the delay acts only as a phase on its one frequency mode.

Evaluated events. We select original test patterns with nonzero counts on at least two of the four bypass-idler rails, identifying emission from multiple sources in the lossless paired-source setting. Selection uses only the original occupations. This gives 63,455 queries across total photon numbers $N = 4, 6, 8, 10, 12$, counting both signal and idler photons. Two-photon events cannot meet the selection criterion. All selected queries are evaluated at every delay, giving 1,015,280 reference calculations.

Comparison and aggregation. The zero-delay input keeps PhOracle's predictions at $\Delta t = 0$ fixed, whereas the delay-aware input uses the adjacency computed at each delay. Both are compared with the same newly computed delayed reference probabilities, using a common positive-finite validity mask without probability clipping. Figure 4(e) pools individual APE values across photon totals and source energies at each delay. Figure 10 separates photon totals and includes 95% intervals from 2,000 whole-circuit bootstrap resamples, stratified by source energy and paired across delays and input variants.

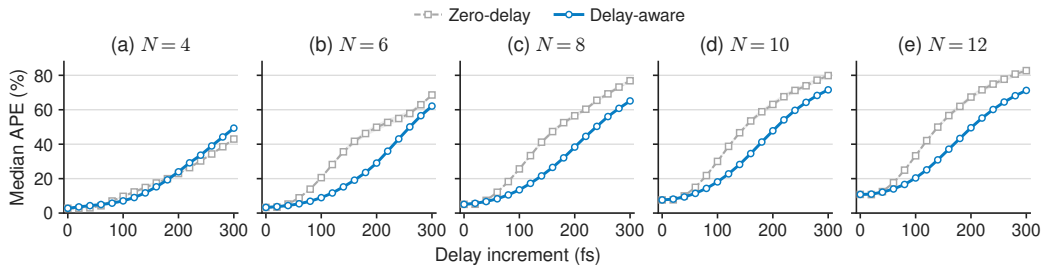

Figure 10: **Temporal mismatch by photon number.** Median APE against delayed reference probabilities for events involving at least two sources. Bands show 95% circuit-bootstrap intervals. The signal-path delays are $0, \Delta t, 2\Delta t, 3\Delta t$.

E.4 Component Ablations

All structural variants use the same data, 75,000-step training exposure, and the selected PhOracle recipe. Each variant retains the checkpoint with the lowest validation Log-MSE. Ablation confidence intervals resample complete circuits with the checkpoints held fixed.

Table 6: **Common component ablations.** Log-MSE ratios are relative to the full model in the first row.

Variant	Params (M)	Step	Val. MSE	Test MSE	Ratio	Median APE (%)
Full PhOracle	5.487	75,000	0.01370	0.01734	1.000	4.56
One recurrent step	5.486	28,500	0.09321	0.1078	6.219	16.57
One step, nine-layer Rail Transformer	12.883	29,000	0.07589	0.09022	5.203	13.98
Query-independent recurrence	5.487	72,500	0.1099	0.1258	7.254	22.43
Global token only	5.444	71,500	0.01800	0.02252	1.299	4.68
Without product normalization	5.487	35,000	0.02552	0.03196	1.843	9.58
Final depth only	5.487	61,500	0.01978	0.02406	1.387	6.30
No $\mathcal{L}_{\text{spacing}}$	5.487	73,000	0.01617	0.02011	1.160	5.13

Recurrent updates. Error decreases with every added update at all photon totals (Figure 11(a)). The two-, three-, and four-update models use the same global batch size and training recipe as the earlier controls, but are trained on one GPU rather than eight. The one-update model with a nine-layer Rail Transformer improves over the three-layer one-update model but remains less accurate than every multi-update model. The full model makes six three-layer Rail Transformer calls, including the initial state evaluation. The single-update control makes two nine-layer calls, giving 18 Transformer-layer applications in either case. The full model additionally performs four more matrix-state updates, so this comparison matches Rail Transformer layer applications rather than total computation.

Query conditioning. Conditioning the matrix-state updates on the requested pattern reduces error across the photon-number range (Figure 11(b)). The query-independent variant removes occupation information from the recurrent state updates, then applies the requested pattern through a readout-only Rail Transformer pass at each depth.

Product normalization and readout. Product normalization has the largest effect among the readout choices tested (Figure 11(c)). The raw-product variant removes only the shared root-mean-square normalization before feature extraction, keeping the readout dimensions unchanged. The global-token-only variant removes the explicit product features and feeds the global token to the same two-hidden-layer readout. Using only the final-depth prediction is also less accurate than the learned mixture of depth-specific predictions. Figure 11(c) uses paired circuit bootstrap samples for the ratio of each variant's Log-MSE to the full model's Log-MSE.

Loss ablation. We retain a spacing weight of 0.10 because it yields a small reduction in average validation Log-MSE compared with MSE-only training.

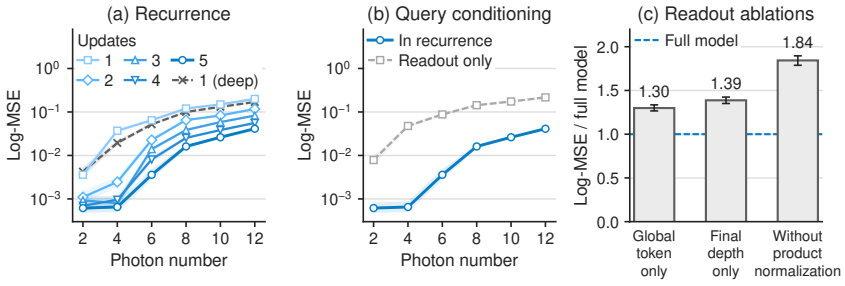

Figure 11: **Contributions of PhOracle's components.** **(a)** One to five recurrent updates with a three-layer Rail Transformer, alongside one update with a nine-layer Rail Transformer (deep). **(b)** Query-conditioned updates versus query-independent updates with conditioning only at readout. **(c)** Using only the global token, restricting readout to the final depth, or removing product normalization. Curves pool ID and pattern-OOD queries at each photon total. Intervals are 95% circuit-bootstrap intervals.

Symmetry checks. The implementation tests apply rail-local detector-basis changes and jointly permute the rails in the adjacency and occupation query. They check that the predictions retain the invariances specified by the architecture.

F QUANTUM-STATE TOMOGRAPHY

We specify the target state, then describe the analyzer bank, least-squares reconstruction, optimization objective, and reference-solver verification. PhOracle-assisted and reference-solver search use the same analyzer bank and reconstruction map.

F.1 TARGET AND STATE REPRESENTATION

Both downstream tasks optimize a frequency-flat $U(5)$ and reconstruct a two-photon state on logical rails 0–3. The target is $(|1010\rangle - |0101\rangle)/\sqrt{2}$. The single-pair task conditions on idler counts $(1, 1, 1, 1)$ and two ancilla photons. The multipair task uses idler counts $(2, 1, 1, 1)$ and three ancilla photons. The idlers occupy rails 4–7 and the ancilla occupies rail 8, leaving two photons on the logical rails in both tasks.

What tomography reconstructs. Quantum-state tomography recovers a density matrix from probabilities measured in several bases. Direct PNR detection determines basis populations but cannot distinguish the Bell-minus state from an incoherent mixture of $|1010\rangle$ and $|0101\rangle$. Passive analyzers rotate the state before detection, making the missing coherences observable as population differences.

Spatial state and exchange symmetry. An ideal single-mode two-photon state on four rails lies in a 10-dimensional symmetric Fock space. With unresolved spectra, bosonic symmetry applies to the joint spatial and spectral state. An antisymmetric spatial component can accompany an antisymmetric spectral component. Tracing out the spectrum can therefore leave both symmetric and antisymmetric spatial components. We represent the reduced spatial density matrix ρ in the 16-dimensional labeled-particle basis $|j, k\rangle$ of $(\mathbb{C}^4)^{\otimes 2}$, where $j, k \in \{0, 1, 2, 3\}$ denote the rail occupied by each photon. Exchange invariance separates symmetric and antisymmetric blocks of dimensions 10 and 6, leaving $10^2 + 6^2 = 136$ real Hermitian components before trace normalization. In this representation the Bell target is

$$|\Psi^-\rangle = \frac{|0, 2\rangle + |2, 0\rangle - |1, 3\rangle - |3, 1\rangle}{2}. \tag{47}$$

F.2 ANALYZER BANK

Banchi et al. (2018) establish multiphoton tomography with linear optics and photon counting and numerically demonstrate the efficiency of Haar-random measurement configurations. Our analyzer bank contains the identity $U_0 = I_4$ and 31 independently drawn Haar-random $SU(4)$ matrices. The bank is sampled once and shared by both tasks and evaluators. We generate the Haar-random matrices by phase-corrected complex QR factorization (Mezzadri, 2007). Removing their determinant phase gives $SU(4)$ matrices without affecting photon-count probabilities.

Analyzer U_s, with $s \in \{0, \ldots, 31\}$, acts on the labeled two-photon space as $\mathcal{U}(U_s) = U_s \otimes U_s$. Each setting measures all 10 two-photon occupation patterns, indexed by $k \in \{1, \ldots, 10\}$. The projector $\mathcal{P}_k$ selects all labeled output states with occupation pattern k, including both label orders when the photons occupy distinct rails. The conditional measurement probability is

$$q_{sk} = \mathrm{Tr}\left[\mathcal{P}_k \, \mathcal{U}(U_s) \rho \mathcal{U}(U_s)^\dagger\right]. \tag{48}$$

Collecting these 320 linear equations gives the measurement matrix $M_{\mathrm{QST}} \in \mathbb{C}^{320 \times 256}$, with one column for each entry of ρ. Its rank is 136 at a relative singular-value cutoff of 10^{-10}, sufficient to identify both exchange-symmetry blocks of ρ.

F.3 LEAST-SQUARES RECONSTRUCTION

Flattening the 32×10 conditional probabilities into $\mathbf{q} \in \mathbb{R}^{320}$ gives $\mathbf{q} = M_{\mathrm{QST}} \, \mathrm{vec}(\rho)$ for a measurement-consistent state. Here vec stacks the rows of a matrix into a vector, and unvec reverses this operation. Independently predicted probabilities need not lie in this 136-dimensional measurement range. We therefore compute the minimum-norm least-squares solution

$$\widetilde{\mathbf{v}}_\rho = M_{\mathrm{QST}}^{+} \mathbf{q}, \qquad \widetilde{\mathbf{v}}_\rho \in \arg\min_{\mathbf{v}} \|M_{\mathrm{QST}} \mathbf{v} - \mathbf{q}\|_2^2, \tag{49}$$

where M_{QST}^{+} is the Moore–Penrose pseudoinverse, precomputed with the same 10^{-10} relative singular-value cutoff. The solution minimizes measurement error and sets unidentifiable null-space components to zero.

The implementation reshapes $\widetilde{\mathbf{v}}_\rho$, averages the result with its conjugate transpose, and normalizes its trace:

$$\widetilde{\rho} = \mathrm{unvec}(\widetilde{\mathbf{v}}_\rho), \qquad \rho_{\mathrm{herm}} = \frac{\widetilde{\rho} + \widetilde{\rho}^\dagger}{2}, \qquad \widehat{\rho} = \frac{\rho_{\mathrm{herm}}}{\max\{\mathrm{Tr}(\rho_{\mathrm{herm}}), \epsilon_P\}}, \qquad \epsilon_P = 10^{-30}. \tag{50}$$

The floor prevents division by a vanishing trace. When it is inactive, the reconstructed matrix has unit trace. We report the relative tomography residual after these operations,

$$r_{\mathrm{QST}} = \frac{\|\mathrm{Re}[M_{\mathrm{QST}}\,\mathrm{vec}(\widehat{\rho})] - \mathbf{q}\|_2}{\|\mathbf{q}\|_2}. \tag{51}$$

The residual measures how closely one reconstructed operator reproduces the complete measurement vector.

Differentiable objective. We adapt the Lie-algebraic Bures optimization of Hartnett et al. (2026), replacing direct access to a conditional Fock-state vector with differentiable QST of the potentially mixed logical state. For unitary parameters $\boldsymbol{\varphi}$, the probabilities, reconstruction, and Bures-angle objective are

$$\begin{aligned} P_{sk}(\boldsymbol{\varphi}) &= P(\mathbf{n}_{sk} \mid A_s[U(\boldsymbol{\varphi})])\,, \\ q_{sk}(\boldsymbol{\varphi}) &= \frac{P_{sk}(\boldsymbol{\varphi})}{\sum_{j=1}^{10} P_{sj}(\boldsymbol{\varphi})}, \\ \widehat{\rho}(\boldsymbol{\varphi}) &= \mathcal{T}_{\mathrm{QST}}(\{q_{sk}(\boldsymbol{\varphi})\}_{s,k})\,, \\ \mathcal{L}_{\mathrm{QST}}(\boldsymbol{\varphi}) &= \arccos\sqrt{\langle\Psi^-|\widehat{\rho}(\boldsymbol{\varphi})|\Psi^-\rangle}. \end{aligned} \tag{52}$$

The nine-entry $\mathbf{n}_{sk}$ combines the fixed herald and ancilla counts with logical outcome k. The adjacency A_s includes analyzer s. The map $\mathcal{T}_{\mathrm{QST}}$ denotes the least-squares reconstruction, Hermitian symmetrization, and trace normalization above. Both search arms differentiate this computation, changing only the evaluator of P_{sk}. The probability-trained PhOracle is frozen throughout this state-level optimization.

For reconstruction from surrogate predictions, Hermitian symmetrization and trace normalization do not enforce positive semidefiniteness, so the Bell overlap $\mathcal{F}_{\text{score}} = \langle\Psi^-|\widehat{\rho}|\Psi^-\rangle$ is a search score rather than a verified state fidelity. We clip this score to $[10^{-7}, 1-10^{-7}]$ before computing the Bures angle. Final fidelities are evaluated from reference probabilities.

F.4 Reference-Solver Verification

Every saved endpoint is reconstructed from its $SU(5)$ coordinates and reevaluated using 320 joint probabilities from the reference solver. We check three properties: positive semidefiniteness of the reconstructed state, agreement with the measurement probabilities, and consistency of the herald probability across analyzers. For analyzer s, let $h_s = \sum_{k=1}^{10} P_{sk}$ be the probability of the fixed herald and ancilla event, summed over all logical outcomes. Rotating the logical measurement basis does not change this event, so its probability should be independent of the analyzer. We measure deviations using

$$r_h = \frac{\sqrt{\frac{1}{32}\sum_{s=0}^{31}(h_s - \bar{h})^2}}{\max\{\bar{h}, \epsilon_P\}}, \qquad \bar{h} = \frac{1}{32}\sum_{s=0}^{31} h_s. \tag{53}$$

Writing $\lambda_{\min}(\widehat{\rho})$ for the smallest eigenvalue, we accept a reference-reconstructed state when $\lambda_{\min}(\widehat{\rho}) \geq -10^{-8}$, $r_{\mathrm{QST}} \leq 10^{-8}$, and $r_h \leq 10^{-8}$. Table 7 reports these diagnostics for the selected circuits before and after refinement and for the best reference-search circuits. The selected surrogate overlap scores are 0.9707 and 0.9751 for the single-pair and multipair tasks, respectively. The table reports independently reference-evaluated fidelities.

Table 7: Physical verification of the selected circuits using reference probabilities. Rescored and refined rows evaluate PhOracle-selected candidates before and after refinement. Reference rows evaluate the best reference-search circuit. Small negative eigenvalues are within the 10^{-8} numerical tolerance.

Task	Stage	Fidelity	$\lambda_{\min}(\widehat{\rho})$	r_{QST}	r_h
Single-pair	Rescored	0.9758	-1.60×10^{-15}	4.68×10^{-15}	1.04×10^{-14}
	Refined	0.9919	-3.84×10^{-16}	5.32×10^{-15}	9.42×10^{-15}
	Reference	0.9938	-1.38×10^{-16}	3.81×10^{-15}	6.81×10^{-15}
Multipair	Rescored	0.9376	-1.16×10^{-15}	3.56×10^{-15}	4.80×10^{-15}
	Refined	0.9943	-1.32×10^{-15}	3.70×10^{-15}	6.25×10^{-15}
	Reference	0.9944	-6.64×10^{-16}	4.26×10^{-15}	6.33×10^{-15}

Clipping during search. Clipping was inactive in all logged optimizer evaluations of the two surrogate trajectories that produced the selected circuits ($0/229$ single-pair and $0/273$ multi-pair iterations). We checked the saved Bures-angle losses against both clipping boundaries, $\arccos\sqrt{1-10^{-7}}$ and $\arccos\sqrt{10^{-7}}$, with a tolerance of 10^{-10} radians.

G CIRCUIT-DISCOVERY DETAILS

We describe candidate selection and refinement, followed by the unitary parameterization, optimizer and stopping rules, and hardware and timing methodology.

G.1 PHORACLE-ASSISTED SEARCH

Gradients of the reconstructed Bell-state objective pass through the frozen PhOracle model to update the unitary (Figure 12). Both arms use three paired starts at each of 25 source energies, holding the source fixed within a search. For each task, the lowest surrogate Bures angle across 75 candidates determines the single PhOracle circuit selected for refinement. The reference-solver search selects its best circuit by the reference-evaluated Bures angle. Reference rescoring evaluates the selected circuit without modification before refinement. Reference-solver search uses the same initializations, objective, and global stopping rule. Both methods optimize conditional Bell fidelity with fixed postselection counts. Heralding probability is neither an optimization objective nor a candidate-selection criterion.

G.2 UNITARY PARAMETERIZATION AND OPTIMIZATION

We parameterize the interferometer by 24 real variables $\boldsymbol{\varphi}$, using a matrix exponential to preserve unitarity:

$$U(\boldsymbol{\varphi}) = \exp\left(i \sum_{a=1}^{24} \varphi_a T_a\right) U_{\text{anchor}}, \qquad \mathrm{Tr}(T_a T_b) = \delta_{ab}. \tag{54}$$

Here $\{T_a\}$ are traceless Hermitian generators whose multiples iT_a form a basis of $\mathfrak{su}(5)$. There are $5^2 - 1 = 24$ real parameters because the overall unitary phase does not affect photon-count probabilities. Each paired comparison draws an initial vector $\boldsymbol{\varphi}$ with coordinates $\varphi_a \sim \mathcal{N}(0, 0.25^2)$ and supplies the same draw to both search methods, with $U_{\text{anchor}} = I_5$. For reference-solver refinement, U_{anchor} is the selected PhOracle unitary and all 24 parameters start at zero.

All arms use Adam with $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and no weight decay. A OneCycle schedule increases the learning rate from $1/25$ of its peak over the first 20% of the maximum step budget, then decays it to $1/(25 \times 10^4)$ of the peak using cosine annealing. Adam's β_1 cycles between 0.85 and 0.95, inversely to the learning rate. The schedule is parameterized by the maximum step budget of each stage and is truncated when the convergence rule fires. Table 8 gives the peak learning rate, minimum and maximum step counts $t_{\min}$ and $t_{\max}$, patience N_{pat}, and minimum improvement ϵ_{conv} for each stage. Each update uses all 320 queries and clips the global gradient norm to 5.0.

Table 8: **Stopping rules and peak learning rates.** Bures-angle thresholds ϵ_{conv} are in radians.

Stage	$t_{\min}$	$t_{\max}$	N_{pat}	ϵ_{conv}	Peak LR
Global search (both arms)	150	500	20	3×10^{-5}	0.02
Reference-solver refinement	30	120	10	3×10^{-5}	0.01

The first evaluated loss sets the comparison value, and the counter starts at zero. A subsequent loss more than ϵ_{conv} below this value replaces it and resets the counter. Otherwise, the counter increases by one. Once at least $t_{\min}$ steps have been evaluated, the stage stops after N_{pat} consecutive evaluations without such an improvement, or on reaching $t_{\max}$. Each stage returns the lowest-loss iterate, including improvements smaller than ϵ_{conv} that do not reset the counter. Refinement starts with a fresh Adam state and OneCycle schedule.

G.3 HARDWARE AND TIMING

The full energy sweep uses a host with two NVIDIA RTX PRO 6000 Blackwell Server Edition GPUs, with one GPU assigned to each search or refinement process. The reported speedup divides the sum of 75 reference-search process times by the sum of 75 PhOracle-search process times plus the time for one reference-solver refinement. These sums measure accumulated per-process elapsed time, rather than the elapsed time of the parallel run. Process times include initialization, source

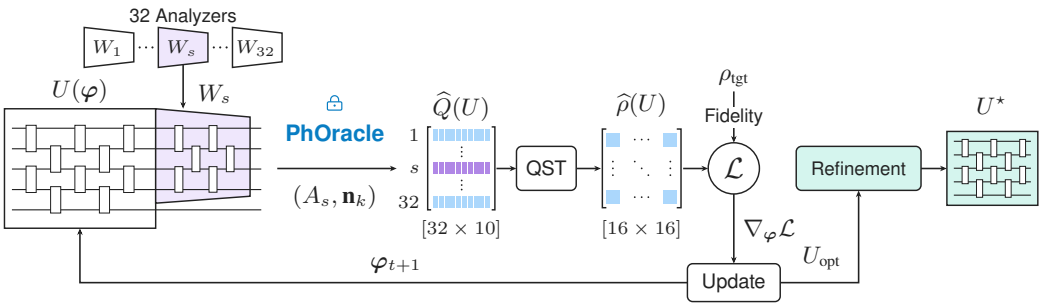

Figure 12: **PhOracle-assisted circuit design.** PhOracle probabilities reconstruct a density matrix whose Bell overlap guides unitary optimization. The selected circuit is rescored, refined, and verified with the reference solver.

preparation, search, endpoint evaluation and diagnostics, and result serialization. Refinement time includes its final reference evaluation. Offline training, dataset generation, and checks shared by both methods before the searches are excluded. Optimization-only timings separately measure the search iterations until the stopping rule returns the best-so-far iterate, with CUDA synchronization at timing boundaries.

Table 9: **Accumulated discovery time in minutes.** Search rows sum 75 runs per task, whereas refinement is performed once. Other process time excludes timed optimizer iterations.

Stage	Single-pair	Multipair
PhOracle search: optimization	33.79	34.64
PhOracle search: other process time	10.65	14.00
Reference-solver refinement: complete process	4.57	8.10
Hybrid total	**49.02**	**56.75**
Reference search: optimization	840.50	2039.25
Reference search: other process time	8.91	12.19
Reference total	**849.42**	**2051.44**