This is the accepted manuscript made available via CHORUS. The article has been published as:

Measuring the similarity of graphs with a Gaussian boson sampler

Maria Schuld, Kamil Brádler, Robert Israel, Daiqin Su, and Brajesh Gupt

Phys. Rev. A **101**, 032314 — Published 11 March 2020

DOI: 10.1103/PhysRevA.101.032314

Measuring the similarity of graphs with a Gaussian Boson Sampler

Maria Schuld,[1,*] Kamil Brádler,[1,†] Robert Israel,[1] Daiqin Su,[1] and Brajesh Gupt[1]

[1] *Xanadu, Toronto, Canada*

(Dated: January 21, 2020)

Gaussian Boson Samplers (GBS) have initially been proposed as a near-term demonstration of classically intractable quantum computation. We show here that they have a potential practical application: Samples from these devices can be used to construct a feature vector that embeds a graph in Euclidean space, where similarity measures between graphs - so called 'graph kernels' - can be naturally defined. This is crucial for machine learning with graph-structured data, and we show that the GBS-induced kernel performs remarkably well in classification benchmark tasks. We provide a theoretical motivation for this success, linking the extracted features to the number of r-matchings in subgraphs. Our results contribute to a new way of thinking about kernels as a quantum hardware-efficient feature mapping, and lead to a promising application for near-term quantum computing.

I. INTRODUCTION

Measuring the similarity of two graphs for practical applications is notoriously difficult. Firstly, there are many different notions of similarity, and practical tasks crucially depend on what property of the graph is exploited in the comparison. Secondly, even the task of determining whether two graphs are exactly the same can be computationally extremely costly. This is due to the fact that a representation of a graph is not unique: Different ways of enumerating its nodes and edges can give rise to the same structure. The complexity of deciding whether two graphs are isomorphic is unknown; neither a polynomial-time algorithm nor NP-completeness proof has been discovered yet [1]. Existing algorithms for graph isomorphism [2] and graph similarity [3] are efficient in practice, but are still costly for large graphs and may require exponential time for some problem instances.

In this paper we suggest the use of quantum hardware to map a graph G to a feature vector which represents G in Euclidean space. Standard distance measures, such as taking the inner product of two feature vectors, then result in a distance measure between graphs mediated by the feature embedding. The quantum device we investigate is a Gaussian Boson Sampling (GBS) setup [4–6]. GBS is a generalization of Boson Sampling [7–10], which has originally been proposed as a classically intractable problem to demonstrate the power of near-term quantum hardware [11]. An optical GBS device prepares a quantum state of M optical modes and counts the photons in each mode. Some of the authors have previously shown how a graph can be encoded into the quantum state of light [12], so that the photon measurement statistics give rise to a complete set of graph isomorphism invariants [13].

Here we extend this result and study the graph similarity measure derived from a GBS device for a practical application, namely for classification for machine learning. Graph-structured data plays an increasingly important role in this field, for example to predict properties of a social media network given a dataset of networks for which the properties are known. In machine learning, a similarity measure between data is called a *kernel*, and lots of methods for pattern recognition – such as support vector machines and Gaussian processes – are built around this concept. Mapping graphs to feature vectors or graph embeddings [14–16] is a well known strategy, and graph kernels from explicit feature vectors [17] have been studied in detail.

The connection between kernel methods for machine learning and quantum computing has recently been made in Refs. [18, 19]. Any positive-definite kernel can be formally understood as the inner product of two feature vectors that represent the data points in a Hilbert space [20]. Hence, the Hilbert space of a quantum system can be interpreted as a feature space, in which a subroutine can compute inner products "coherently". By using measurement samples from the quantum hardware to construct low-dimensional feature vectors that can be stored and further processed on a classical computer, we follow a different, even more minimalistic route to define a "quantum feature map", and ultimately a quantum kernel. The advantage in using quantum hardware this way is that device performs a combinatorial computation that is very resource-intense – possibly even intractable – for classical computers. In fact, we show that the GBS feature map is related to a class of classical graph kernels which count subgraphs [21], but instead of only considering subgraphs of constant size, the sampling statistics can reveal information on all possible subgraphs, as well as subgraphs constructed from copying nodes and their edges. The resulting features contain information about the number of r-matchings of the original graph. Numerical experiments show that graph kernels from a GBS-induced feature map can outperform classical graph kernels in classification task for small standard benchmark datasets, results that can be further improved by using displaced light modes.

This paper is organized as follows: In Section II we present the GBS graph similarity framework, including a

* maria@xanadu.ai

† kamil@xanadu.ai

recap of previous results. Section III investigates the GBS-extracted features more closely, and Section IV presents numerical experiments to motivate the relevance of the findings in practice.

II. TURNING GBS SAMPLES INTO FEATURES

An optical Gaussian Boson Sampler is a device where a special quantum state (a so-called *Gaussian state*) is prepared by the *optical squeezing* of M *displaced* light modes, followed by an *interferometer* of beamsplitters. A pure Gaussian state is fully described by a covariance matrix $\sigma \in \mathbb{R}^{2M\times 2M}$ as well as a displacement vector $\mathbf{d} \in \mathbb{R}^{2M}$ [22]. Photon number resolving detectors count the photons in each mode.

In this section we describe the mathematical details of the quantum hardware-induced feature map (see also Figure 1), summarizing what has been described in Brádler et al. [13], and adding the effect of displacement as well as a further step of turning samples to feature vectors. The scheme works for *simple graphs*, i.e. undirected graphs without self-loops or multiple edges. While edge weights can be treated on the same footing as unweighted edges, we leave the inclusion of categorical edge labels or node labels for future studies. Mindful of readers from fields other than quantum optics we will only highlight some important aspects of Gaussian Boson Sampling and refer to Refs. [4, 6, 12] for more detail.

A. Encoding graphs into the GBS device

As outlined in [12], a quantum state prepared by a GBS device can encode a graph $G = (V, E)$ with an adjacency matrix A of entries A_{ij} that are one if the edge (i, j) exists in G and zero else. The entries of A can also represent continuous "edge weights" that denote the strength of a connection. In the latter case we will speak of a "weighted adjacency matrix".

In order to associate A with the symmetric, positive definite $2M$-dimensional covariance matrix of a Gaussian state of M modes, we have to construct a "doubled adjacency matrix"

$$\tilde{A} = c\begin{pmatrix} A & 0 \\ 0 & A \end{pmatrix} = c(A \oplus A), \tag{1}$$

where the rescaling constant c is chosen so that $0 < c < 1/s_{\max}$, and $s_{\max}$ is the maximum singular value of A [12, 13].[23] For simplicity we will always rescale all adjacency matrices with a factor $1/(s_{\max}^{\{G\}} + 10^{-8})$ where $s_{\max}^{\{G\}}$ is the largest singular value among all graphs in the data set under consideration. As a result we will assume that $c = 1$ and $\tilde{A} = A \oplus A$ can be encoded into a GBS device. We call this the "doubled encoding strategy".

The matrix $\tilde{A}$ can now be associated with a quantum state's covariance matrix σ by setting the squeezing as well as the beamsplitter angles of the interferometer so that

$$\sigma = Q - \mathbb{1}/2, \text{ with } Q = (\mathbb{1} - X\tilde{A})^{-1}, X = \begin{pmatrix} 0 & \mathbb{1} \\ \mathbb{1} & 0 \end{pmatrix}. \tag{2}$$

$\mathcal{M}_{\lvert\mathbf{n}\rvert,\Delta_s}$	$O_{\mathbf{n}}$	$\mathbf{n}$	$\lvert\mathbf{n}\rvert$	$G_{\mathbf{n}}$	$\mathrm{Haf}(A_{\mathbf{n}})$
$\mathcal{M}_{0,\Delta_0}$	$O_{[0,0,0]}$	$[0,0,0]$	0		0
$\mathcal{M}_{1,\Delta_1}$	$O_{[1,0,0]}$	$[1,0,0]$	1		0
		$[0,1,0]$			0
		$[0,0,1]$			0
$\mathcal{M}_{2,\Delta_1}$	$O_{[1,1,0]}$	$[1,1,0]$	2		1
		$[1,0,1]$			1
		$[0,1,1]$			1
$\mathcal{M}_{2,\Delta_2}$	$O_{[2,0,0]}$	$[2,0,0]$	2		0
		$[0,2,0]$			0
		$[0,0,2]$			0
$\mathcal{M}_{3,\Delta_1}$	$O_{[1,1,1]}$	$[1,1,1]$	3		0
$\mathcal{M}_{3,\Delta_2}$	$O_{[2,1,0]}$	$[2,1,0]$	3		0
		$[2,0,1]$			0
		$[1,2,0]$			0
		$[1,0,2]$			0
		$[0,2,1]$			0
		$[0,1,2]$			0
$\mathcal{M}_{3,\Delta_3}$	$O_{[3,0,0]}$	$[3,0,0]$	3		0
		$[0,3,0]$			0
		$[0,0,3]$			0

TABLE I. Events $\mathcal{M}_{|\mathbf{n}|,\Delta_s}$, orbits $O_{\mathbf{n}}$, photon events $\mathbf{n}$, total photon number $|\mathbf{n}|$, extended induced subgraph $G_{\mathbf{n}}$ (indicated by red/black nodes and edges) and Hafnian haf$(A_{\mathbf{n}})$ of a fully connected simple graph of three nodes up to $|\mathbf{n}|_{\max} = 3$. The difference between orbits and events only becomes apparent for higher photon evens (i.e., $[2, 2, 0, 0]$ and $[2, 1, 1, 0]$ are in different orbits but the same event). Note that the red nodes are not mutually connected.

B. Sampling photon counting events

After embedding A via $\tilde{A}$ into the quantum state of the GBS, each measurement of the photon number resolving detectors returns a photon event $\mathbf{n} = [n_1, \ldots, n_M]$, with $n_i \in \mathbb{N}$ indicating the number of photons measured in the i-th mode. Assuming for now that the displacement

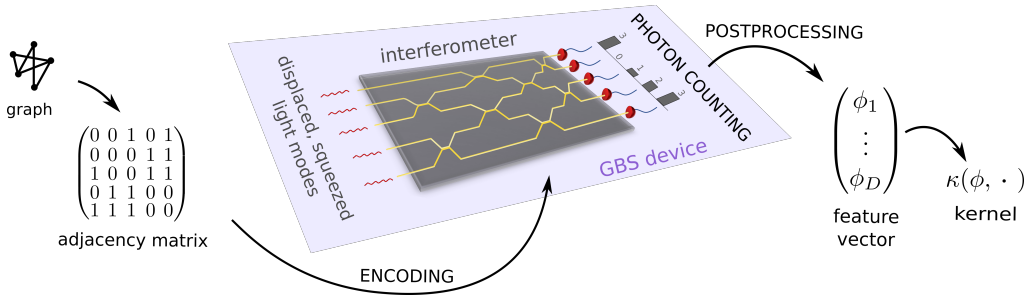

FIG. 1. Idea of the quantum hardware-induced feature map. The adjacency matrix of a graph gets encoded into the Gaussian state of the light modes by tuning the squeezing and interferometer parameters. The features are defined as the probability of detecting certain classes of photon counting events. To extract the probabilities from the device, a number of samples of photon counting events is generated and the relative frequencies of the different classes determined.

$\mathbf{d}$ is zero, the probability of measuring a given photon counting event is

$$p(\mathbf{n}) = \frac{1}{\sqrt{\det(Q)}\,\mathbf{n}!}\,\mathrm{haf}^2(A_\mathbf{n}), \tag{3}$$

where $\mathbf{n}! = n_1!n_2!\cdots n_M!$.

Let us go through this nontrivial equation bit by bit. The Hafnian haf() is a matrix operation similar to the determinant or permanent. For a general symmetric matrix $C \in \mathbb{R}^N \times \mathbb{R}^N$ with matrix elements $C_{u,v}$ it reads

$$\mathrm{haf}(C) = \sum_{\pi \in P_N^{\{2\}}} \prod_{(u,v)\in\pi} C_{u,v}. \tag{4}$$

Here, $P_N^{\{2\}}$ is the set of all $N!/((N/2)!2^{N/2})$ ways to partition the index set $\{1,2,\ldots,N\}$ into $N/2$ unordered pairs of size 2, such that each index only appears in one pair. The Hafnian is zero for odd N. As an example, for the index set $\{1,2,3,4\}$ we have $P_4^{\{2\}} = \{(1,2),(3,4)\},\{(1,3),(2,4)\},\{(1,4),(2,3)\}$.

If C is interpreted as an adjacency matrix containing the edges of a graph, the set $P_N^{\{2\}}$ contains edge-sets of all possible perfect matchings on G. A perfect matching is a subset of edges such that every node is covered by exactly one of the edges. The Hafnian therefore sums the products of the edge weights in all perfect matchings. If all edge weights are constant, it simply counts the number of perfect matchings in G (see also Figure 2). Note that in Eq. (3) we used the fact that for real and symmetric A, $\mathrm{haf}(\tilde{A}) = \mathrm{haf}(A \oplus A) = \mathrm{haf}^2(A)$. In other words, the doubled encoding strategy leads to a square factor which will play a profound role in the quantum feature map we are aiming to construct.

Eq. (3) does not depend on the Hafnian of the adjacency matrix A, but on a matrix $A_\mathbf{n}$. $A_\mathbf{n}$ contains n_j duplicates of the jth row and column in A. If $n_j = 0$, the jth row/column in A does not appear in $A_\mathbf{n}$. Effectively, this constructs a new graph $G_\mathbf{n}$ from A according to the following rules (see also Table I):

1. If all $n_j, j = 1,\ldots,M$ are one (i.e., each detector counted exactly one photon), $A_\mathbf{n} = A$.

2. If some n_j are zero and others one (i.e., these detectors report no photons), $A_\mathbf{n}$ describes an *induced subgraph* $G_\mathbf{n}$ of G, in which nodes that correspond to detectors with zero count were deleted together with any edge that connected them to other nodes.

3. If some n_j are larger than one (i.e., these detectors count more than one photon), $A_\mathbf{N}$ describes what we call an *extended induced subgraph* in which the corresponding nodes and all their connections are duplicated n_j times.

In short, the probability of a photon event to be measured by the GBS device is proportional to the square of the (weighted) number of perfect matchings in a -possibly extended - induced subgraph of the graph encoded into the interferometer.

Computing the Hafnian of a general matrix is in complexity class #P, and formally reduces to the task of computing permanents [24]. If no entry in the matrix is negative, efficient approximation heuristics are known, although their success is only guaranteed under specific circumstances [25, 26].

C. The effect of displacement

The Gaussian Boson Sampling setup underlying Eq. (3) consists of squeezing and interferometers. But a Gaussian quantum state can also be manipulated by a third operation: displacement. Displacement changes the mean of the M-mode Gaussian state while leaving the covariance matrix (and therefore the encoding strategy) as before. A non-zero mean changes Eq. (3) in an interesting, but non-trivial manner.

Without going into the details [6], if considering nonzero displacement, instead of summing over $P_N^{\{2\}}$ in Eq. (4), we have to sum over $P_N^{\{1,2\}}$, or the set of partitions of the index set $\{1,\ldots,N\}$ into subsets of size *up to* 2. For the index set $\{1,2,3,4\}$, we had

$$P_4^{\{2\}} = \{(1,2),(3,4)\},\{(1,3),(2,4)\},\{(1,4),(2,3)\},$$

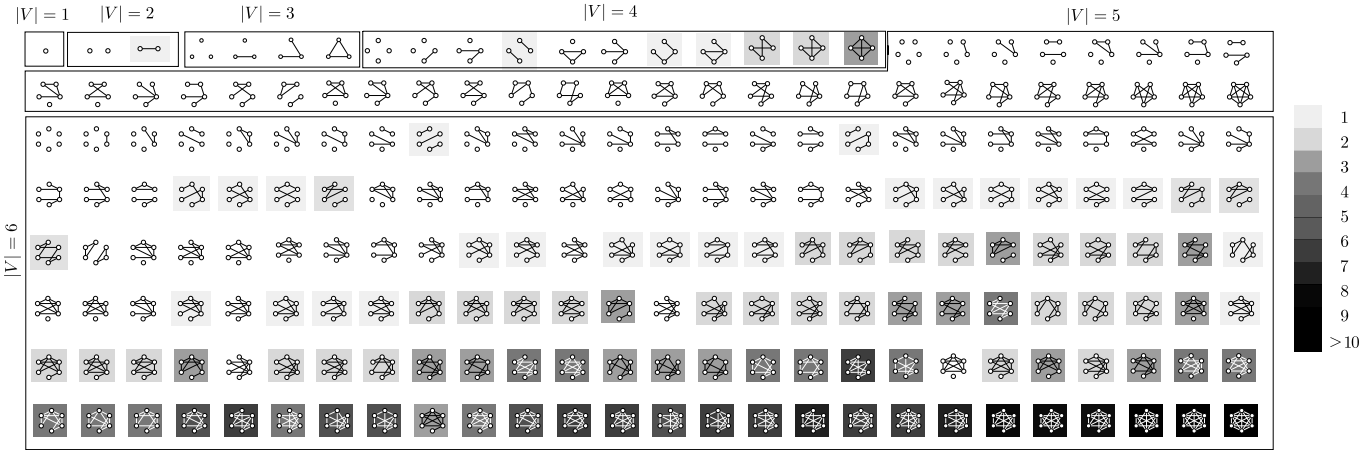

FIG. 2. All non-isomorphic graphs up to size $|V| = 6$ and the number of perfect matchings they contain (grey shading scale).

which now becomes

$$P_4^{\{1,2\}} = \{\{(1,2),(3),(4)\},\{(1,3),(2),(4)\},\{(1,4),(2),(3)\},\{(2,3),(1),(4)\},\{(2,4),(1),(3)\},\{(3,4),(1),(2)\},\{(1,2),(3,4)\},\{(1,3),(2,4)\},\{(1,4),(2,3)\},\{(1),(2),(3),(4)\}$$

Instead of the Hafnian in Eq. (3), we therefore get a mixture of Hafnians of $A_{\mathbf{n}}$'s submatrices (stemming from the pairs) and other factors (stemming from the size-1 sets).

Assume that displacement is applied to both the $\hat{x}$ and $\hat{p}$ quadratures of each mode, described by a $2M$-dimensional displacement vector $\mathbf{d} = (d_1, \ldots, d_M, d_1^*, \ldots, d_M^*)^T$. The effect on Eq. (3) is as follows. Let Q the $2M \times 2M$ matrix from Eq. (2), and $\mathbf{b} = \mathbf{d}^\dagger Q^{-1}$. We get the novel expression (for a derivation, see Appendix A)

$$p(\mathbf{n}) = \alpha \left[\sum_{n=0}^{M} \sum_{\{i_1..i_n\} \subseteq \mathcal{I}_M} b_{i_1} \ldots b_{i_n} \mathrm{haf}(A_{\mathbf{n}-\{i_1..i_n\}}) \right]^2 \tag{5}$$

with

$$\alpha = \frac{e^{-\frac{1}{2}\mathbf{d}^\dagger Q^{-1}\mathbf{d}}}{\sqrt{\det(Q)}\,\mathbf{n}!},$$

where $\mathcal{I}_{2M}$ is the index set $\{1, \ldots 2M\}$. In this notation we assume $\{i_1, \ldots, i_0\} = \{\}$ and $b_{i_1} \ldots b_{i_0} = 1$. The "reduced" Hafnians of the form $A_{\mathbf{n}-\{i,j\}}, A_{\mathbf{n}-\{i,j,k,l\}} \cdots$ are constructed by "deleting" rows and columns $\{i,j\},\{i,j,k,l\},...$ in $\tilde{A}_{\mathbf{n}}$. The expression in the brackets of Eq. (5) is also known as a "loop Hafnian" of a matrix $\tilde{A}_{\mathbf{n}}$ that carries $b_1, ..., b_{2M}$ on its diagonal [27].

One can see that displacement explores substructures of extended subgraphs, adding another layer of "resolution" to the photon number distribution. An important effect of displacement is that $p(\mathbf{n})$ for odd total photon numbers $|\mathbf{n}|$ is not necessarily zero any more, since the sum in Eq. (5) contains Hafnians of even-sized subgraphs.

D. Turning samples into features

The basic idea of how to turn samples of photon counting events into feature vectors is to associate the probability of a certain measurement result with a feature. To estimate the probability of measurement outcomes, one divides the number of times a result has been measured by the total number of measurements. However, if we used the probabilities of photon events $p(\mathbf{n})$ directly as features, we would face a very fast – more precisely, a doubly factorial – explosion of the number of features with the total number of photons, while almost all events become vanishingly unlikely for realistic amounts of squeezing. In practice we will truncate the total number of photons at a fixed value k and discard all measurement results with $|\mathbf{n}| > k$ in the construction of the feature vector, but even then the sampling task quickly becomes unfeasible.

We therefore define the probability of certain *types* of photon events as features, thereby "coarse-graining" the probability distribution. As a compromise between experimental feasibility and expressive power, we consider two different coarse-graining strategies here. The first one follows Brádler et al.'s [13] suggestion to coarse-grain the distribution of photon counting events by summarizing them to sets called *orbits* (see Table I). An orbit $O_{\mathbf{n}} = \{\mathrm{perm}(\mathbf{n})\}$ contains permutations of the detection event $\mathbf{n}$. For example, $[2,1,1,0]$ is in the same orbit as $[0,1,2,1]$, but not $[2,2,0,0]$. The photon counting event $\mathbf{n}$ in the index is therefore an arbitrary "representative" of the photon counting events in an orbit. The probability of detecting a photon counting event of orbit $O_{\mathbf{n}}$ is given by the sum of the individual probabilities,

$$p(O_{\mathbf{n}}) := \sum_{\mathbf{n} \in O_{\mathbf{n}}} p(\mathbf{n}). \tag{6}$$

The number of orbits $O_{\mathbf{n}}$ containing events of up to k photons in total is equal to the number of ways that the integers of $1, ..., k$ can be partitioned into a sum of at most M terms. In practice we usually have $k \ll M$,

in which case there are $2, 4, 7, 12, 19, 30, 45, 67$ orbits for $k = 1, \ldots, 8$, respectively [28]. In a real GBS setup, the energy is finite and high photon counts therefore become very unlikely [29].

The second post-processing strategy builds on top of the first, and summarizes orbits to *events* $\mathcal{M}_{|\mathbf{n}|,\Delta_s}$, where

$$\Delta_s = \big\{\mathbf{n} : \sum_i n_i = |\mathbf{n}|, (\forall i)(n_i \leq s), (\forall \mathbf{n} \exists n_i \in \mathbf{n})(n_i = s)\big\}.$$

In words, a event contains all orbits of $|\mathbf{n}|$ photons, which have at least one detector counting s photons, but no detector counts more than s photons (see also Table I). The probability of detecting an event from a event is given by

$$p(\mathcal{M}_{|\mathbf{n}|,\Delta_s}) := \sum_{\mathbf{n}\in\Delta_s} p(O_{\mathbf{n}}). \tag{7}$$

From here on, when using event features, we refer to the GBS as "GBS$^+$".

It is interesting to estimate how many samples are needed to estimate a feature vector. In Ref [21] we find that we can approximate a probability distribution of D possible outcomes, with probability at most δ that the sum of absolute values of the errors in the empirical probabilities of the outcomes is ϵ or more, using

$$S = \left\lceil \frac{2(\log(2)D + \log(\frac{1}{\delta}))}{\epsilon^2} \right\rceil$$

samples. For orbits up to $k = 8$ photons, there are $D = 67$ features. Setting $\epsilon = 0.05$ and $\delta = 0.05$ and assuming a perfect GBS device, we need $39,550$ samples. Since current-day photon number resolving detectors can accumulate about 10^5 samples of photon counting events per second [30], it takes in principle only a fraction of a second for the orbit probabilities to be estimated by the physical hardware. The number of samples does not grow with the graph size, but of course the GBS device itself grows linearly in the number of nodes.

Hardware implementations of Gaussian Boson Samplers are rapidly advancing, but mode numbers larger than 10 with a tuneable interferometer and squeezing are still a huge experimental challenge. In this paper we therefore resort to simulations on classical computers. While for non-negative adjacency matrices, efficient approximation algorithms to calculate Hafnians are known [25, 26, 31, 32], sampling from distributions that depend on Hafnians is still a topic of active research [33], and to ensure that the results are not influenced by approximation errors we will use exact calculations here. This limits the scope of the experiments to graphs of the order of 25 nodes.

E. Constructing a similarity measure

Summarizing the above, the feature map implemented by a GBS device maps a graph to a feature vector, $G \to \mathbf{f} \in \mathbb{R}^D$, where the entries $f_i, i = 1, \ldots, D$ of $\mathbf{f}$ are the probabilities of detecting certain types of photon events that we called orbits and events,

$$f_i = p(O^i_{\mathbf{n}}), \text{ or } f_i = p(\mathcal{M}^i_{|\mathbf{n}|,\Delta_s}), \tag{8}$$

and the probability of the i'th (meta-)orbit is fully defined by Eqs. (6) and (7) (while ordering in the feature vector does not matter).

Assuming that the maximum number k of photons we consider is smaller or equal to the number of detectors, or $k \leq M$ for all graphs, the size D of the feature vector is solely determined by k, which is a hyperparameter of the feature map. Another hyperparameter is the displacement that can be applied to the light modes. We will assume here that the displacement applied to all modes is a constant value d.

Once constructed, the feature vectors can be used for various applications. In the context of machine learning, they can be directly fed into neural network classifiers. Here we are interested in constructing a similarity measure or kernel that computes the similarity between two graphs G and G'. A standard choice is to use the feature vectors in a 'linear' and 'rbf' kernel (with a hyperparameter δ)

$$\kappa_{\text{lin}}(G, G') = \langle \mathbf{f}, \mathbf{f}' \rangle,$$
$$\kappa_{\text{rbf}}(G, G') = \exp\left(-\frac{||\mathbf{f} - \mathbf{f}'||^2}{2\delta^2}\right),$$

both of which are well known to be positive semi-definite so that the results of kernel theory apply to the "GBS kernel" constructed here.

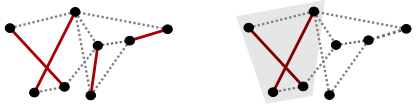

FIG. 3. Example of a perfect matching (left) and a 2-matching (right). The 2-matching is at the same time a perfect matching of the subgraph highlighted in grey.

III. THE GBS GRAPH FEATURES

In this section we will analyze the features of the first post-processing strategy in more detail; we discuss their intimate relation to the coefficients of a graph property called a "matching polynomial", the relation of photon event probabilities to higher-order moments of multivariate normal distributions, the connection between the GBS and graphlet sampling kernel, and we finally discuss the devastating effect of photon loss on the features.

A. Single-photon features and r-matchings

It turns out that the probabilities of 'single-photon' orbits (i.e., each detector counts either zero or one photon)

are related to a graph property called the "matching polynomial" of G [34–36],

$$\mu(G) = \sum_{r=0}^{\lceil M/2 \rceil} (-1)^r m(G,r) x^{M-2r}. \tag{9}$$

The coefficients $m(G,r)$ of the matching polynomial count the number of r-matchings or "independent edge sets" in G – sets of r edges that have no vertex in common (see Figure 3). In the language of Hafnians, the r matching can be written as $m(G,r) = \sum_{\mathbf{n} \in O_{[1,\dots,1,0,\dots]}} \mathrm{haf}(A_\mathbf{n})$ (where $[1,\dots,1,0,\dots]$ contains $2r$ single photon detections). Hence, if it were not for the square of the Hafnian in Eq. (3), the probability $p(O_\mathbf{n})$ of a single-photon orbit would be proportional to a $|\mathbf{n}|/2$-matching $m(G, |\mathbf{n}|/2)$ of G. The square gives rise to a new object

$$g(G,r) = \sum_{\mathbf{n} \in O_{[1,\dots,1,0,\dots]}} \mathrm{haf}^2(A_\mathbf{n}).$$

Replacing m with g in Eq. (9) leads to a new type of polynomial $\gamma(G)$ which we call a *GBS polynomial*.

This definition opens up a range of interesting questions, for example whether the GBS polynomial has advantages over a standard polynomial, or how multi-photon events and displacement fits into this interpretation. We will investigate these questions in separate works.

An interesting observation for the context of machine learning occurs for the feature corresponding to orbit $O_{[1,1,0,\dots]}$ (see for example Table I). Since there are only two options – the two nodes are connected and have therefore exactly one perfect matching, or they are not and have none – the square does not have any effect, and the probability of the orbit is proportional to the number of 1-matchings of this graph, which is in turn equal to its number of edges. Hence, we have that $p(O_{[1,1,0,\dots]}) \propto |E|$, and the hardware natively returns an "edge counting" feature.

B. Higher-order moments

The probability of measuring a given photon counting event $\mathbf{n} = [n_1, \dots, n_M]$ can also be interpreted from a slightly different, more physically motivated viewpoint. The M nodes of a graph can be associated with M random variables drawn from a multivariate normal distribution $N(\xi, \Sigma)$, where the covariance matrix Σ corresponds to the doubled adjacency matrix $\tilde{A}$, and ξ is the mean vector related to displacement via $\xi = Q^{-1}\mathbf{d}^\dagger$. The higher-order moments $E[X_1^{(1)} \dots X_1^{(n_1)} \dots X_M^{(1)} \dots X_M^{(n_M)}]$ of this distribution are proportional to $\mathrm{haf}(A_\mathbf{n})$, which in turn is related to the probability of a photon event via Eq. (3). This result follows from *Isserlis' theorem* [37], which decomposes the higher order moments into sums of products of covariances $E[X_a X_b]$. In short, the GBS device turns a graph into a multivariate normal distribution and samples from its moments.

Using this picture, the first-order moments of the 'graph-induced distribution' correspond to photon events of the form $[1, 0, \dots]$ and their probability is indeed proportional to the mode means as apparent from Eq. (5). The second-order moments correspond to photon events of the form $[1, 1, 0, \dots]$ and their probability is proportional to the entries of the adjacency matrix – the edge weights. Consistent with this observation, we stated before that orbits with 2 non-zero detectors "measure" the edge count of a graph.

While the doubled encoding strategy as well as the presence of multi-photon events somewhat obscure interpretations of features in terms of r-matchings and higher-order moments, we found in numerical experiments not reported in this paper that they can be a blessing in disguise, making very similar graphs distinguishable by smaller maximum photon numbers k.

C. Comparison to Graphlet Sampling kernel

Counting subgraphs in a larger graph is a concept used in various classical graph kernels. Graphlet Sampling kernels [21] bear the most striking similarity to GBS feature maps, since the features count how often graphlets of size $|V| = 3, 4, 5, \dots$ appear in a graph G. In the language developed here we can express the feature f_g which counts graphlet g via

$$f_g \propto \sum_{\mathbf{n} \in O_{[1,..,1,0,\dots]}} \mathbb{1}_{g \cong G_\mathbf{n}}, \tag{10}$$

using an indicator function $\mathbb{1}_{g \cong G_\mathbf{n}}$ that is one if graphlet g is isomorphic to the subgraph $G_\mathbf{n}$ and zero else, as well as the orbit represented by $[1, .., 1, 0, \dots]$ counting $|V|$ single photons. In comparison, rewriting Eq. (8) in a similar way, the GBS features are

$$f_i = f_{\mathbf{n}_i^*} \propto \sum_{\mathbf{n} \in O_{\mathbf{n}^*}} \left(\sum_{g \in \mathcal{P}^{|\mathbf{n}|}} \mathbb{1}_{g \cong G_\mathbf{n}} \right)^2, \tag{S4}$$

where $\mathcal{P}^{|\mathbf{n}|}$ is the set of all perfect matchings of size $|\mathbf{n}|$. As a result, instead of counting graphlets, the GBS feature map sums squares of perfect matching counts in graphlets. Also, GBS feature map does not restrict the size of the graphlet probed.

D. Errors due to photon loss

One of the main sources of errors in a realistic GBS device is a photon loss in the linear interferometer, and we demonstrate here that loss is a serious problem for applications of a GBS for graph similarity as proposed in this paper. Methods of dealing with this kind of errors will be discussed in upcoming work. Here we show the effect of the loss on the coarse-grained probabilities with a numerical example.

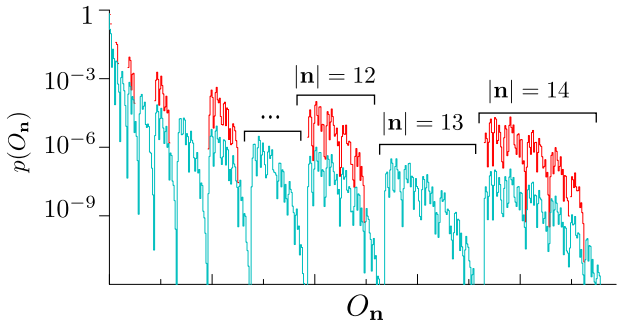

FIG. 4. Coarse-grained probability $p(O_{\mathbf{n}})$ where G is an random unweighted graph on 10 vertices. We compare the lossless scenario (red) with a lossy case (blue) of 3dB photon loss ($\nu = 0.5$ in (11)). The squeezing is the same in both cases and its maximal value is 6.2dB. The orbits O_n on the x axis are first ordered according to the total photon number $|n|$, and then in ascending order of the single-detector photon numbers, i.e. $[0,0,0...], [1,0,0...], [1,1,0...], [2,0,0...], ...$ A major tick on the x-axis counts 100 orbits. The gaps of the total odd photon numbers for the red curve indicate zero probability which is consistent with the single modes squeezed states occupying the even subspace of the Hilbert space of Fock states.

The effect of loss is described by the action of the lossy bosonic channel on a pure covariance matrix σ resulting in

$$\sigma(\nu) = (1-\nu)\sigma + \frac{\nu}{2}\mathbb{1}_M, \tag{11}$$

where $\nu = 1-\eta$ and η is the overall transmissivity. One way of viewing this is that the matrix $\tilde{A}$ from Eq. ((1)) does not have the block-diagonal structure $c(A \oplus A)$ any more, but is of the form

$$\tilde{C} = X_{2M}\Big(\mathbb{1}_{2M} - W^{-1}\mathrm{diag}\left[\frac{\Lambda_1 - 1}{\Lambda_1\nu - 1}, \ldots, \frac{\Lambda_{2M} - 1}{\Lambda_{2M}\nu - 1}\right] W\Big),$$

where $W X_{2M}\tilde{A}W^{-1} = \mathbf{\Lambda}$ is the eigendecomposition of $X_{2M}\tilde{A}$. Figure 4 shows the effect of this loss model on the probability distribution $p(O_{\mathbf{n}})$ over orbits for a random unweigthed graph G on ten vertices. It is apparent that loss introduces errors in the distribution, populating orbits which have a zero probability in the zero-displacement case, and distorting the remaining probabilities significantly. In the remainder of the paper we will consider only a lossless GBS device, but remark herewith that loss mitigation strategies are crucial for practical applications of GBS feature maps.

IV. NUMERICAL EXPERIMENTS

Finally, we provide some numerical results to investigate the GBS graph kernel in practice. Benchmarks suggest that it is well competitive to standard "classical" graph kernels, at least in the hypothetical case of a perfect device. We furthermore show that displacement may improve classification accuracy by shifting weight into the higher-order orbits, and that orbits with photon numbers smaller or equal to 2 contribute most to the result.

A. Benchmarking

To benchmark the GBS feature map, we use a setup that has become a standard in testing graph kernels: A C-Support Vector Machine (SVM) with a precomputed kernel. The test accuracies in Table II are obtained by running 10 repeats of a double 10-fold cross-validation. The inner fold extracts the best model by adjusting the C-parameter of the SVM – which controls the penalty on misclassifications – via grid search between values $[10^{-4}, 10^3]$, and the best model is then used to get the accuracy of the test set in the outer cross-validation loop. The GBS feature vectors were used in conjunction with a 'rbf' kernel κ_{rbf}.

For the GBS graph kernel, we chose a gentle displacement of $d = 0.25$ on every mode and $k = 6$, leading to 30-dimensional feature vectors. We used exact simulations based on the hafnian library [38]. These are computationally very expensive, which is why we only consider small datasets. Three classical graph kernels are benchmarked for comparison: The Graphlet Sampling kernel [21] (GS) with maximum graphlet size of $k = 5$ and 5174 samples drawn, the Random Walk kernel [39] (RW) with fast computation and a geometric kernel type, and the Subgraph Matching kernel (SM) [40]. The three classical kernels were simulated using Python's *grakel* library [41].[42]

The datasets are taken from the repository of the Technical University of Dortmund [43] (see Figure 5). They are briefly described in Appendix B. Preprocessing of the benchmarking datasets includes these three steps:

1. *Graph selection*: Graphs which have less than 6 or more than 25 nodes are excluded to keep the feature vectors constant and to limit the time of simulations. The share of excluded graphs is displayed in Figure (3) in the main paper, and ranges from 5% to 55%.

2. *Labels and attributes*: Potential node labels, node attributes and edge attributes are ignored, and the edge weights were binarized as described in Appendix B.

3. *Rescaling*: The final (weighed or unweighed) adjacency matrix is divided by a normalization constant $c = 1/(\lambda_{\max}^{\{G\}} + 10^{-8})$ that is slightly larger than the largest eigenvalue $\lambda_{\max}^{\{G\}}$ of any adjacency matrix in the dataset, as explained in Section 2.1. Note that for most datasets used here, $\lambda_{\max}^{\{G\}} \approx 3$, and the squeezing becomes unphysically large for actual experiments. However, c only rescales the features by known factors which are the same for every feature vector. In practice one can therefore choose a more convenient c parameter which negotiates

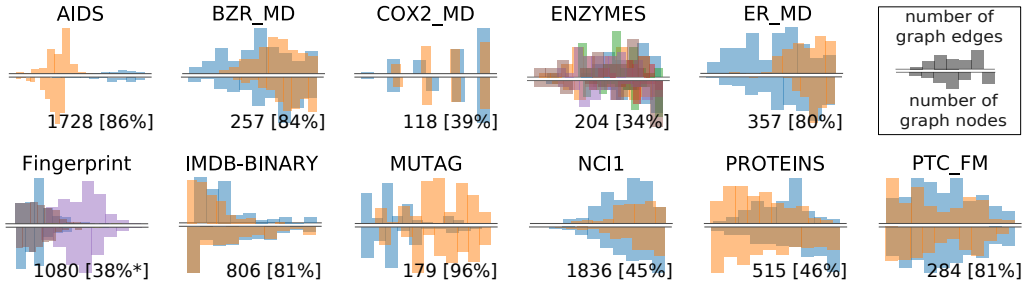

FIG. 5. Dimensionless histograms of node and edge numbers of graphs in the benchmark datasets, to visualize the relative shapes of the class distributions (plotted in different colours). The number of graphs as well as its percentage with respect to the original data are shown below each plot. *Some classes in 'Fingerprint' were excluded due to insufficient samples of small graphs.

Dataset	GBS ($d_{0.0}$)	GBS ($d_{0.25}$)	GBS$^+$ ($d_{0.0}$)	GBS$^+$ ($d_{0.25}$)	GS	RW	SM
AIDS	99.60 ± 0.05	$\mathbf{99.62} \pm 0.03$	99.58 ± 00.06	99.61 ± 0.05	98.44 ± 0.09	56.95 ± 7.99	79.20 ± 0.68
BZR_MD	62.73 ± 0.71	62.13 ± 1.44	62.01 ± 1.43	$\mathbf{63.16} \pm 2.11$	60.60 ± 1.77	49.88 ± 3.74	61.90 ± 1.21
COX2_MD	44.98 ± 1.80	50.11 ± 0.97	57.84 ± 4.04	57.89 ± 2.62	55.04 ± 3.33	57.72 ± 3.26	$\mathbf{66.94} \pm 1.22$
ENZYMES	22.29 ± 1.60	28.01 ± 1.83	25.72 ± 2.60	$\mathbf{40.42} \pm 2.02$	35.87 ± 2.19	21.13 ± 1.91	36.70 ± 2.83
ER_MD	70.36 ± 0.78	70.41 ± 0.47	71.01 ± 1.26	$\mathbf{71.05} \pm 0.83$	65.65 ± 1.06	68.75 ± 0.53	68.21 ± 0.99
FINGERPRINT	65.42 ± 0.49	$\mathbf{65.85} \pm 0.36$	66.19 ± 00.84	66.26 ± 4.29	64.10 ± 1.52	47.69 ± 0.21	47.14 ± 0.62
IMDB-BIN	64.09 ± 0.34	$\mathbf{68.71} \pm 0.59$	68.14 ± 0.71	67.60 ± 0.75	68.37 ± 0.62	66.38 ± 0.21	out of time*
MUTAG	$\mathbf{86.41} \pm 0.33$	85.58 ± 0.59	85.64 ± 0.78	84.46 ± 0.44	81.08 ± 0.93	83.02 ± 1.08	83.14 ± 0.24
NCI1	$\mathbf{63.61} \pm 0.00$	62.79 ± 0.00	63.59 ± 0.17	63.11 ± 0.93	49.96 ± 3.27	52.36 ± 2.63	51.36 ± 1.88
PROTEINS	$\mathbf{66.88} \pm 0.22$	66.14 ± 0.48	65.73 ± 0.69	66.16 ± 0.76	65.91 ± 1.29	56.27 ± 1.23	63.03 ± 0.84
PTC_FM	53.84 ± 0.96	52.45 ± 1.78	59.14 ± 1.72	56.25 ± 2.04	$\mathbf{59.48} \pm 1.95$	51.97 ± 2.68	54.92 ± 2.94

TABLE II. Mean test accuracy of the Support Vector Machine with different datasets and different graph kernels, with the standard deviation between 10 repetitions of the double cross-validation. GS, RW, and SM are three standard classical graph kernels described in the text. GBS refers to the postprocessing strategy of associating orbit probabilities with features, while GBS$^+$ summarises some orbits to events (see text).*$Runtime > 20$ *days.*

between squeezing levels in reach of hardware and high enough photon numbers to resolve the features.

All datasets were chosen *before* the first experiments were run, to avoid a post-selection bias in favour of the GBS kernel.

As Table II shows, the GBS kernel performs well and outperforms the other methods visibly for MUTAG and NCI1, while still leading for AIDS, BZR_MD, ER_MD, FINGERPRINT and PROTEINS. Displacement increases the performance of the GBS kernel significantly for COX2_MD, ENZYMES and IMDB-BIN, but not for other data sets. The GBS kernel does well on datasets where the distribution of node and edge numbers differs strongly between classes. However, we confirmed that excluding the 'edge counting features' $[1, 1, 0..], [2, 2, 0..], \ldots$ does not influence classification performance. While the graph size is considered by the GBS kernel, it seems to be only one of many properties that enters the notion of similarity.

B. Displacement and feature importance

The hyperparameters of the GBS and GBS$^+$ graph kernels are the constant displacement d which adiministered to each node, as well as the maximum photon number k. Since simulations restrict the value of k at this stage, we focus on the effect of displacement, using the orbit-features (i.e., the GBS kernel). Displacement can change the similarity measure significantly. For example, comparing graphs of size $|V| = 3$, one finds that the fully disconnected graph is closer to the fully connected graph than a graph with two edges for $d = 1$, but vice versa for $d = 0$.

Figure 6 uses the example of IMDB-BIN and MUTAG to investigate the GBS or "orbit" features for $d = 0, d = 0.25$ and $d = 1$. The feature averages show that the general distribution of the feature vector is similar for both classes, but still visually distinguishable.[44] Consistent with the theory, increasing displacement shifts the features towards higher-order orbits, and populates features that are zero when $d = 0$. Features associated with orbits $[1, 1, 0, ...], [1, 1, 1, 1, 0...]$ and $[1, 1, 1, 1, 1, 1]$, as well as $[2, 1, 0, ...]$ and $[2, 1, 1, 1, 0, ...]$ seem to be particularly important in the support of principal components, and get high weights when training a perceptron on the GBS features. Where displacement renders them nonzero, uneven orbits such as $[1, 1, 1, 0, ...], [1, 1, 1, 1, 1, 0...]$ follow suit. During our investigations we confirmed that drop-

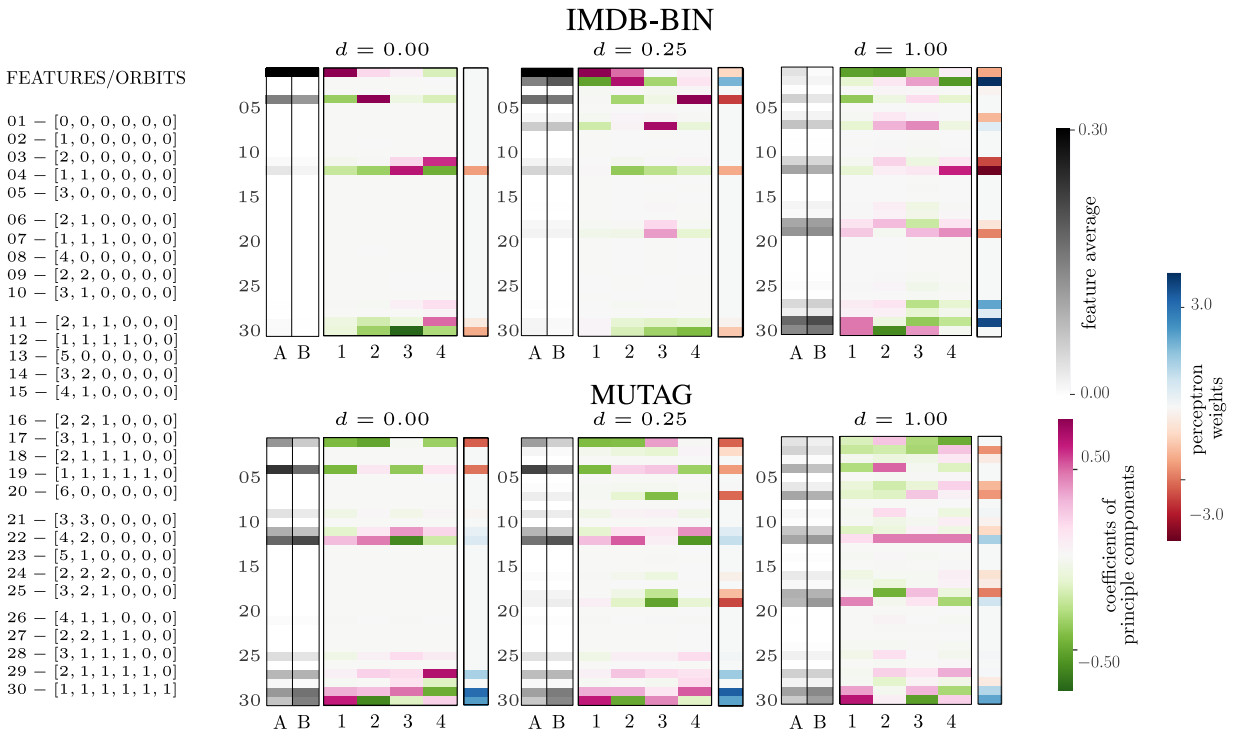

FIG. 6. Three measures for feature importance for IMDB-BINARY (top row) and MUTAG (bottom row) using $k = 6$ and for $d = 0, d = 0.25$ and $d = 1$. The $3 + 3$ heatmaps consist of three columns each. The leftmost column (gray color map) shows the average of each feature for the two different classes, here labeled A and B. The center column shows the coefficients with which each feature contributes to the four first principal components in the PCA analysis. The third column shows the weights which a perceptron attributes to each feature when trained to classify the target labels.

ping features with high single-detector photon numbers did not have a huge influence on classification. Consistent with the results from Table II, MUTAG has 'richer' features for $d = 0$ than IMDB-BIN for classification with a perceptron, an advantage that IMDB-BIN equalizes with growing displacement.

The feature analysis suggests that features related to subgraphs of all sizes (here 1 to 6) are important for the classification results, and that duplication of a single node in the subgraphs may be beneficial – a feature that Graphlet Sampling kernels do not explore. The effect of displacement varies with the dataset, and d should therefore be kept as a hyperparameter for model selection.

V. CONCLUSION

We proposed a new type of feature extraction strategy for graph-structured data based on the quantum technique of Gaussian Boson Sampling. We suggested that the success of the method is related to the fact that such a system samples from distributions that are related to useful graph properties. For classical machine learning, this method presents a potentially powerful extension to the gallery of graph kernels, each of which has strengths on certain data sets. For quantum machine learning, this proposes the first application of a "quantum kernel".

A lot of questions are still open for further investigation, for example regarding the role and interpretation of displacement, how GBS performs with weighted adjacency matrices, how node and edge labels can be considered, as well as whether the feature vectors are useful in combination with other methods such as neural networks. We expect that the rapid current development of numeric GBS samplers as well as quantum hardware will help answering these questions in the near future.

ACKNOWLEDGEMENTS

We thank Christopher Morris and Nicolas Quesada for valuable advice, as well as the authors of Python's GraKel library. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725.

[1] Johannes Kobler, Uwe Schöning, and Jacobo Torán, *The graph isomorphism problem: its structural complexity* (Springer Science & Business Media, 2012).

[2] Brendan D McKay *et al.*, *Practical graph isomorphism* (Department of Computer Science, Vanderbilt University Tennessee, USA, 1981).

[3] Swarnendu Ghosh, Nibaran Das, Teresa Gonçalves, Paulo Quaresma, and Mahantapas Kundu, "The journey of graph kernels through two decades," Computer Science Review **27**, 88–111 (2018).

[4] Craig S Hamilton, Regina Kruse, Linda Sansoni, Sonja Barkhofen, Christine Silberhorn, and Igor Jex, "Gaussian boson sampling," Physical Review Letters **119**, 170501 (2017).

[5] AP Lund, A Laing, S Rahimi-Keshari, T Rudolph, Jeremy L OBrien, and TC Ralph, "Boson sampling from a gaussian state," Physical Review Letters **113**, 100502 (2014).

[6] Regina Kruse, Craig S Hamilton, Linda Sansoni, Sonja Barkhofen, Christine Silberhorn, and Igor Jex, "A detailed study of Gaussian Boson Sampling," arXiv preprint arXiv:1801.07488 (2018).

[7] Max Tillmann, Borivoje Dakić, René Heilmann, Stefan Nolte, Alexander Szameit, and Philip Walther, "Experimental boson sampling," Nature Photonics **7**, 540 (2013).

[8] Matthew A Broome, Alessandro Fedrizzi, Saleh Rahimi-Keshari, Justin Dove, Scott Aaronson, Timothy C Ralph, and Andrew G White, "Photonic boson sampling in a tunable circuit," Science **339**, 794–798 (2013).

[9] Justin B Spring, Benjamin J Metcalf, Peter C Humphreys, W Steven Kolthammer, Xian-Min Jin, Marco Barbieri, Animesh Datta, Nicholas Thomas-Peter, Nathan K Langford, Dmytro Kundys, *et al.*, "Boson sampling on a photonic chip," Science **339**, 798–801 (2013).

[10] Andrea Crespi, Roberto Osellame, Roberta Ramponi, Daniel J Brod, Ernesto F Galvao, Nicolo Spagnolo, Chiara Vitelli, Enrico Maiorino, Paolo Mataloni, and Fabio Sciarrino, "Integrated multimode interferometers with arbitrary designs for photonic boson sampling," Nature Photonics **7**, 545 (2013).

[11] Scott Aaronson and Alex Arkhipov, "The computational complexity of linear optics," in *Proceedings of the forty-third annual ACM symposium on Theory of computing* (ACM, 2011) pp. 333–342.

[12] Kamil Brádler, Pierre-Luc Dallaire-Demers, Patrick Rebentrost, Daiqin Su, and Christian Weedbrook, "Gaussian boson sampling for perfect matchings of arbitrary graphs," Physical Review A **98**, 032310 (2018).

[13] Kamil Brádler, Shmuel Friedland, Josh Izaac, Nathan Killoran, and Daiqin Su, "Graph isomorphism and Gaussian boson sampling," arXiv preprint arXiv:1810.10644 (2018).

[14] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang, "Network representation learning: A survey," IEEE Transactions on Big Data (2018).

[15] Palash Goyal and Emilio Ferrara, "Graph embedding techniques, applications, and performance: A survey," Knowledge-Based Systems **151**, 78–94 (2018).

[16] Aditya Grover and Jure Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2016) pp. 855–864.

[17] Nils Kriege, Marion Neumann, Kristian Kersting, and Petra Mutzel, "Explicit versus implicit graph feature maps: A computational phase transition for walk kernels," in *Data Mining (ICDM), 2014 IEEE International Conference on* (IEEE, 2014) pp. 881–886.

[18] Maria Schuld and Nathan Killoran, "Quantum machine learning in feature Hilbert spaces," Physical Review Letters **122**, 040504 (2019).

[19] Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, Aram W Harrow, Abhinav Kandala, Jerry M Chow, and Jay M Gambetta, "Supervised learning with quantum-enhanced feature spaces," Nature **567**, 209 (2019).

[20] Bernhard Scholkopf and Alexander J Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond* (MIT press, 2001).

[21] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Artificial Intelligence and Statistics* (2009) pp. 488–495.

[22] 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**, 621 (2012).

[23] As long as it fulfills the above inequality, c can be treated as a hyperparameter of the feature map, which may also be influenced by hardware constraints since it relates ultimately to the amount of squeezing required.

[24] Leslie G Valiant, "The complexity of computing the permanent," Theoretical Computer Science **8**, 189–201 (1979).

[25] Alexander Barvinok, "Approximating permanents and hafnians," arXiv preprint arXiv:1601.07518 (2016).

[26] Mark Rudelson, Alex Samorodnitsky, Ofer Zeitouni, *et al.*, "Hafnians, perfect matchings and gaussian matrices," The Annals of Probability **44**, 2858–2888 (2016).

[27] Nicolás Quesada, "Franck-condon factors by counting perfect matchings of graphs with loops," The Journal of Chemical Physics **150**, 164113 (2019).

[28] See also A000070 in the Online Encyclopedia of Integer Sequences, `https://oeis.org/A000070`.

[29] The energy of a Gaussian quantum state, and hence the average photon number, is determined by the squeezing and displacement operations.

[30] VD Vaidya, B Morrison, LG Helt, R Shahrokhshahi, DH Mahler, MJ Collins, K Tan, J Lavoie, A Repingon, M Menotti, *et al.*, "Broadband quadrature-squeezed vacuum and nonclassical photon number correlations from a nanophotonic device," arXiv preprint arXiv:1904.07833 (2019).

[31] Piotr Sankowski, "Alternative algorithms for counting all matchings in graphs," in *Annual Symposium on Theoretical Aspects of Computer Science* (Springer, 2003) pp. 427–438.

[32] Alexander Barvinok, "Polynomial time algorithms to approximate permanents and mixed discriminants within a simply exponential factor," Random Structures & Algorithms **14**, 29–61 (1999).

[33] Nicolás Quesada and Juan Miguel Arrazola, "The classical complexity of gaussian boson sampling," arXiv preprint arXiv:1908.08068 (2019).

[34] E.J Farrell, "An introduction to matching polynomials," Journal of Combinatorial Theory, Series B **27**, 75–86 (1979).

[35] Chris D. Godsil and Ivan Gutman, "On the theory of the matching polynomial," Journal of Graph Theory **5**, 137–144 (1981).

[36] Ole J Heilmann and Elliott H Lieb, "Theory of monomer-dimer systems," in *Statistical Mechanics* (Springer, 1972) pp. 45–87.

[37] Leon Isserlis, "On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables," Biometrika **12**, 134–139 (1918).
[38] Andreas Björklund, Brajesh Gupt, and Nicolás Quesada, "A faster hafnian formula for complex matrices and its benchmarking on the titan supercomputer," arXiv preprint arXiv:1805.12498 (2018).
[39] Thomas Gärtner, Peter Flach, and Stefan Wrobel, "On graph kernels: Hardness results and efficient alternatives," in *Learning Theory and Kernel Machines* (Springer, 2003) pp. 129–143.
[40] Nils Kriege and Petra Mutzel, "Subgraph matching kernels for attributed graphs," arXiv preprint arXiv:1206.6483 (2012).
[41] Giannis Siglidis, Giannis Nikolentzos, Stratis Limnios, Christos Giatsidis, Konstantinos Skianis, and Michalis Vazirgiannis, "Grakel: A graph kernel library in python," arXiv preprint arXiv:1806.02193 (2018).
[42] Experiments were run on IBM's cloud platform using four 2.8GHz Intel Xeon-IvyBridge Ex (E7-4890-V2-PentadecaCore) processors with 15 CPU cores each, as well as on Oak Ridge's Titan supercomputer.
[43] Kristian Kersting, Nils M. Kriege, Christopher Morris, Petra Mutzel, and Marion Neumann, "Benchmark data sets for graph kernels," (2016), http://graphkernels.cs.tu-dortmund.de.
[44] Standarization of the feature vectors to emphasize their mutual differences improved classification accuracy in some cases, but deteriorated it in others.
[45] Kaspar Riesen and Horst Bunke, "Iam graph database repository for graph based pattern recognition and machine learning," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)* (Springer, 2008) pp. 287–297.
[46] Jeffrey J Sutherland, Lee A O'brien, and Donald F Weaver, "Spline-fitting with a genetic algorithm: A method for developing classification structure- activity relationships," Journal of Chemical Information and Computer Sciences **43**, 1906–1915 (2003).
[47] Karsten M Borgwardt and Hans-Peter Kriegel, "Shortest-path kernels on graphs," in *Data Mining, Fifth IEEE International Conference on* (IEEE, 2005) pp. 8–pp.
[48] Jeroen Kazius, Ross McGuire, and Roberta Bursi, "Derivation and validation of toxicophores for mutagenicity prediction," Journal of Medicinal Chemistry **48**, 312–320 (2005).
[49] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt, "Weisfeiler-lehman graph kernels," Journal of Machine Learning Research **12**, 2539–2561 (2011).
[50] Christoph Helma, Ross D. King, Stefan Kramer, and Ashwin Srinivasan, "The predictive toxicology challenge 2000–2001," Bioinformatics **17**, 107–108 (2001).

Appendix A: Adding displacement

Equation (5) in the main text can be derived from the following expression reported in [6],

$$\begin{aligned} p(\mathbf{n}) &= \frac{e^{-\frac{1}{2}\mathbf{d}^\dagger Q^{-1}\mathbf{d}}}{\sqrt{\det(Q)}\,\mathbf{n}!}\left[\mathrm{Haf}(\tilde{A}_{\mathbf{n}}) + \sum_{i\neq j}^{2M} b_i b_j \mathrm{Haf}(\tilde{A}_{\mathbf{n}-\{i,j\}}) + \cdots + \prod_{j}^{2M} b_j\right], \\ &= \frac{e^{-\frac{1}{2}\mathbf{d}^\dagger Q^{-1}\mathbf{d}}}{\sqrt{\det(Q)}\,\mathbf{n}!}\sum_{n=0}^{M}\sum_{\{i_1..i_{2n}\}\subseteq\mathcal{I}_{2M}} b_{i_1}\cdot\cdots\cdot b_{i_{2n}}\mathrm{Haf}(\tilde{A}_{\mathbf{n}-\{i_1,\ldots,i_{2n}\}}), \end{aligned} \tag{A1}$$

where the notation is consistent with Equation (5), and $\tilde{A}$ is the "doubly encoded" adjacency matrix. The equivalence of Equations (5) and (A1) is similar to the square rule $\mathrm{Haf}(\tilde{A}) = \mathrm{Haf}(A)^2$ in the regime of zero displacement.

To show the equivalence, one uses the fact that for $\tilde{A}$ being a direct sum $A \oplus A$, the index set $i_1, \ldots, i_{2n} \in \mathcal{I}_{2M}$ that Eq. (A1) sums over can be divided into two index sets: $j_1, \ldots, j_s$ which contains all s indices from the 'first subspace' (i.e., the first M dimensions) of $\tilde{A}$, and $k_1, \ldots, k_{s'}$ containing the s' indices from the 'second subspace', and $s + s' = 2n$. The fact that $\mathrm{Haf}(A \oplus B) = \mathrm{Haf}(A)\mathrm{Haf}(B)$, allows us to express the Hafnian of reduced versions of $\tilde{A}_{\mathbf{n}}$ as a product of reduced versions of matrix $\tilde{A}_{\mathbf{n}}$,

$$\mathrm{Haf}(\tilde{A}_{\mathbf{n}-\{i_1,\ldots,i_{2n}\}}) = \mathrm{Haf}(A_{\mathbf{n}-\{j_1,\ldots,j_s\}})\mathrm{Haf}(A_{\mathbf{n}-\{k_1,\ldots,k_{s'}\}}).$$

Altogether, we can therefore write:

$$
\begin{aligned}
p(\mathbf{n}) &\propto \sum_{n=0}^{M} \sum_{\{i_1..i_{2n}\}\subseteq\mathcal{I}_{2M}} b_{i_1}\dots b_{i_{2n}}\mathrm{Haf}(\tilde{A}_{\mathbf{n}-\{i_1..i_{2n}\}}) \\
&= \sum_{s,s'=0}^{M} \sum_{\{j_1..j_s\}\subseteq\mathcal{I}_M} \left(b_{j_1}\dots b_{j_s}\mathrm{Haf}(A_{\mathbf{n}-\{j_1..j_s\}})\right) \sum_{\{k_1..k_{s'}\}\subseteq\mathcal{I}_M} \left(b_{k_1}\dots b_{k_{s'}}\mathrm{Haf}(A_{\mathbf{n}-\{k_1..k_{s'}\}})\right) \\
&= \left(\sum_{n=0}^{M} \sum_{\{i_1..i_n\}\subseteq\mathcal{I}_M} b_{i_1}\dots b_{i_n}\mathrm{Haf}(A_{\mathbf{n}-\{i_1..i_n\}})\right)^2 .
\end{aligned}
$$

Appendix B: Data sets

Here we give further information on the datasets used in the numerical experiment. Except from IMDB-BINARY and Fingerprint, all datasets are from the chemical domain. In COX_MD, ER_MD, MUTAG and PTC_FM, nodes represent atoms, and edges represent different kinds of bonds. The edges were translated into binary connections via the following key: 0 - no chemical bond, 1 - single bond/double bond/triple bond/aromatic bond. In all remaining datasets, the representation is described below.

- **AIDS** - Each graph represents a chemical compound which the graph label marks as anti-HIV active or not. Nodes represent atoms, and edges represent different kinds of covalent bonds. The edges were translated into binary connections via the following key: 0 - no chemical bond 1 - valence of zero, one or two. See also `https://wiki.nci.nih.gov/display/NCIDTPdata/AIDS+Antiviral+Screen+Data` and [45].
- **BZR_MD** - Each graph represents a benzodiazepine receptor ligand, and the graph labels report in vitro binding affinities above a fixed threshold [46].
- **COX2_MD** - Each graph represents a cyclooxygenase-2 inhibitor, while the graph labels indicate in vitro activities against a human recombinant enzyme [46].
- **ENZYMES** - Nodes represent higher-level secondary structure elements and are connected by an edge if they are neighbours in the enzyme's amino acid sequence, or if they are amongst the three nearest neighbours of each structure in space [47]. Node and edge attributes were ignored. The label corresponds to the Enzyme Commission number, indicating which chemical reactions they catalyze.
- **ER_MD** - Each graph represents an estrogen receptor ligand, and the label reports sur-threshold binding affinity to over 1000 other compounds [46].
- **Fingerprint** - The graphs were extracted [45] from images of fingerprints released by the US NIST institute (`https://www.nist.gov/itl/iad/image-group/nist-special-database-302`). Nodes are put at ending points and bifurcation points of the fingerprint patterns, as well as at regular intervals between those points. Edges correspond to the physical distance between points. Graph labels identify the individuals to which a fingerprint belongs. Only graphs of the three dominant individuals or classes $0, 4, 5$ were considered, since the other classes did not contain a sufficient number of samples after small-graph sub-selection.
- **IMBD-BINARY** - A graph corresponds to a network of co-starring in movies. Nodes are actors, while edges indicate whether (1) or not (0) they appeared in a movie of a certain genre together. The graph labels indicate the genre (action movies and romances) [43].
- **MUTAG** - Each graph represents a chemical compound, whith nodes indicating atoms and edges their mutual covalent bonds [45, 48]. The graph labels distinguish the compounds with respect to their mutagenic properties.
- **NCI1** - Each graph represents a chemical compound, labelled by its activity against non-small cell lung cancer andovarian cancer cell lines [49].
- **PROTEINS** - The graphs correspond to proteins from the Protein Data Bank (`http://www.rcsb.org/pdb/`) [45, 47] and are labeled according to their Enzyme Commission number, indicating which chemical reactions they catalyze.
- **PTC_FM** - Each graph represents a chemical compound, while the label indicates carcinogenicity on rodents [50].