Manuscript: Quantum Graph Kernels, Symmetries, and Speedups

Jue Xu*

July 1, 2022

Abstract

The kernel methods for machine learning and the quantum analogues are reviewed. Moreover, both theoretic and heuristic quantum advantages in machine learning are investigated, mainly in terms of structures (symmetries) of problems.

Contents

1 Introduction **2**
1.1 Preliminary and Notation . . . 2
1.2 Background: SVM and kernel trick . . . 2

2 Graph Kernels and Quantum Random Walks **4**
2.1 Diffusion kernel of a pair of vertices . . . 4
2.2 Continuous-time quantum random walk . . . 5
2.3 Graph kernels of a pair of graphs . . . 5
2.4 Discrete-time quantum random walk . . . 6
2.5 Relations, difference, and examples . . . 6

3 Quantum Advantages and Speedups **9**
3.1 Related works . . . 9
3.2 Provable quantum speedups . . . 11
3.3 Heuristic quantum advantages for learning physics systems . . . 13

4 Experiments **13**
4.1 Datasets and benchmark . . . 13

5 Discussion and Conclusion **14**

References **14**

A Machine Learning and Group Theory [TODO] **17**
A.1 Machine learning . . . 17
A.2 Group theory and symmetries . . . 17

B Symmetries in physics **17**
B.1 Lagrangian formalism . . . 17
B.2 Symmetries in physics with Lagrangians . . . 18

*juexu@cs.umd.edu

1 Introduction

With the remarkable success of machine learning and potential (exponential) advantages of quantum computing [ref], the combination of these two areas attracts much attention. Many quantum versions of machine learning techniques [RML14] [LMR14] [CCL19] are developed and exhibtis substantial (meaningful) speedups, but some quantum advantages are heuristic and strongly depend on the assumptions of models [Tan21]. Since several quantum kernel tricks are coined [Hav+19] [SK19] and (rigorous) advantages are demonstrated in [Gli+21] [LAT21], we focus on quantum graphs kernel based on quantum random walk. The (rigorous) quantum advantages in this context are analyzed in terms of symmetries by theoretical group methods [Kon08] [Ben+20]. On the other hand, the (empirical) connections between quantum machine learing and quantum physics are revealed from the perspective of symmetry.

1.1 Preliminary and Notation

In this work, we restrict ourself to supervised learning (mainly SVM), where we are given a set of labeled data for training to predict labels of new data. The (classical) training data is a set of m data points $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$ where each data point is a pair $(\mathbf{x}, y)$. Normally, the input $\mathbf{x} := (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$ is a vector where d is the number of *features* and its *label* $y \in \Sigma$ is a scalar with some discrete set Σ of alphabet/categories. For simplicity, we assume $\Sigma = \{-1, 1\}$ (binary classification).

Notations: a graph $G = (V, E)$ with vertices V and edges E; a group $\mathbb{G}$ with a subgroup $\mathbb{H}$. The hats on the matrices such as $\hat{A}$, $\hat{H}$ emphasize that they play the roles of operators.

For specific purpose, we use different basis (representations) for quantum states. One is the computational basis $\{|z\rangle\}$ with $z \in [2^n]$ where n is the number of qubits, while the other useful one is the binary representation of computational basis $\{|\mathbf{x}\rangle \equiv |x_1, x_2, \ldots, x_n\rangle\}$ with $x_j \in \{0, 1\}$. For simplicity, we let $N \equiv 2^n$ and $|\mathbf{0}\rangle \equiv |0^n\rangle \equiv |0\rangle^{\otimes n}$ if no ambiguity.

1.2 Background: SVM and kernel trick

Support vector machine (SVM) is one of popular *supervised learning* techniques [CV95]. By optimizing certain objective function with training data, SVM finds a *hyperplane* in the input space which is able to separates (classifies) new data well. (see more details in the appendix) However, input data are not always linearly separable, such as the left one in Fig. 1. So, the *kernel trick* (method) is developed: mapping the input data to higher dimension by a feature map such that the data are linearly separatable in this high dimensional feature space (see Fig. 1 for the intuition).

Definition 1 (feature map)**.** The *feature map* is a function (mapping) $\Phi(\mathbf{x}) : \Omega? \to \mathcal{H}_{\mathcal{K}}$ from a low dimensional space (non-linearly) to a high dimensional Hilbert space $\mathcal{H}$ which is commonly referred to as the *feature space*.

Definition 2 (kernel function)**.** A *kernel function* (mapping) $\mathcal{K} : \mathcal{X} \times \mathcal{X} \to \mathcal{H}$, is defined as inner product

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{H}} \tag{1}$$

w.r.t a feature map $\Phi(\mathbf{x})$. A function $\mathcal{K}$ is a valid kernel if and only if? it corresponding Gram matrix $K_{\mathbf{x},\mathbf{x}'} := \mathcal{K}(\mathbf{x}, \mathbf{x}')$ is symmetric and positive semi-definite. A matrix $K \in \mathbb{R}^{n \times n}$ is *positive semi-definite* (p.s.d) if

$$\forall \mathbf{x} \in \mathbb{R}^n, \mathbf{x}^\intercal K \mathbf{x} \geq 0. \tag{2}$$

Or equivalently, all eigenvalues of K are non-negative. (Mercer's condition...)

Definition 3 (inner product)**.** An *inner product* is a map $\langle \cdot, \cdot \rangle : \mathcal{V} \times \mathcal{V} \to \mathbb{F}$ where $\mathcal{V}$ is a vector space and $\mathbb{F}$ is a number field. And $\forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{V}, a, b \in \mathbb{F}$, the map satisfies the following three properties:

- **Conjugate symmetry**: $\langle \mathbf{x}, \mathbf{y} \rangle = \overline{\langle \mathbf{y}, \mathbf{x} \rangle}$

- **(Bi)Linearity**: $\langle a\mathbf{x} + b\mathbf{y}, \mathbf{z}\rangle = a\langle \mathbf{x}, \mathbf{z}\rangle + b\langle \mathbf{y}, \mathbf{z}\rangle$
- **Positive-definiteness**: $\langle \mathbf{x}, \mathbf{x}\rangle \geq 0$ with equality only when $\mathbf{x} = \mathbf{0}$.

In classical machine learning, we assume $\mathcal{V}$ is $\mathbb{R}^n$ and $\mathbb{F}$ is $\mathbb{R}$. The inner product gives rise to the (l_2) *norm* $\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x}\rangle}$, and a distance metric $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|$. dot product

Euclidean vector space $\mathbb{R}^n$ with dot product is an inner product space,. While in the context of quantum mechanics, the inner product is replaced by Dirac notation $\langle \mathbf{x}|\mathbf{x}'\rangle$ and complex field $\mathbb{C}$.

Definition 4 (Hilbert space)**.** A *Hilbert space* $\mathcal{H}$ is a vector space induced by an inner product such that the inner product yields a complete metric space[?].

Since inner product can be understood as a 'similarity' metric between two vectors[?], a kernel is a type of similarity measure between two data points in the high-dimensional feature space.

Definition 5 (Informal)**.** Any continuous, symmetric, positive definite (kernel function) has corresponding Hilbert space $\mathcal{H}$ called the *Reproducing Kernel Hilbert Space* (RKHS) with the *reproducing property*

$$f(\mathbf{x}) = \langle f(\cdot), \mathcal{K}(\cdot, \mathbf{x})\rangle \tag{3}$$

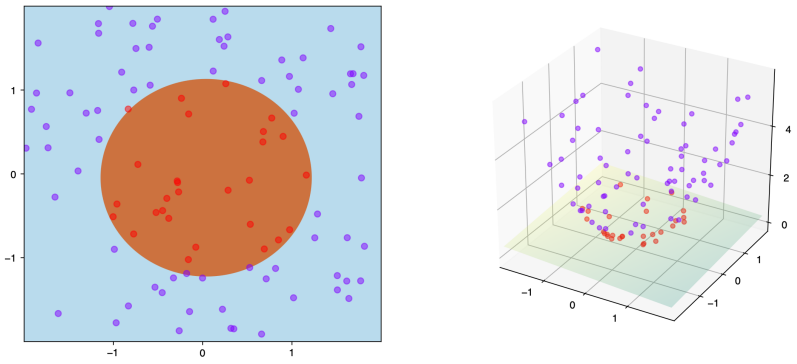

Figure 1: kernel trick idea: SVM with kernel given by $\phi(\mathbf{x}(x_1, x_2)) = (x_1, x_2, x_1^2 + x_2^2)$ and thus $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \mathbf{x} \cdot \mathbf{x}' + \|\mathbf{x}\|^2\|\mathbf{x}'\|^2$. The training points are mapped from a 2-dimensional to a 3-dimensional space where a separating hyperplane can be easily found. (from Wikipedia: Kernel method)

Example 1. Some common (popular) kernels with input space $\mathbb{R}^d$ (dot product):

- *polynomial kernel*: $\mathcal{K}(\mathbf{x}, \mathbf{x}') = (c + \mathbf{x} \cdot \mathbf{x}')^\nu$. Take $d = 2, \nu = 2$ as an example, the corresponding feature map is $\Phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2c}x_1, \sqrt{2c}x_2, c)^\intercal$.
- *Gaussian (radial basis) kernel*:

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) \tag{4}$$

  with the feature map $\Phi(\mathbf{x})? = (\frac{1}{\sqrt{0!}}e^{-\mathbf{x}^2}\mathbf{x}^0, \ldots, \frac{1}{\sqrt{n!}}e^{-\mathbf{x}^2}\mathbf{x}^n, \ldots)$ (infinite dimensions)

Remark 1. Note that the Hilbert space associated with the Gaussian RBF kernel has infinite dimension, but the kernel may be readily computed for any pair of data points. Since both polynomial kernel and Gaussian kernel depends only on the inner product $\mathbf{x} \cdot \mathbf{x}'$, no explicit calculation for the feature maps is required.

Although the development of kernel methods was driven by the empirical success of supervised learning of vector-valued data or image data, in many domains, such as chemo- and bioinformatics, (social) network analysis, data are not naturally interpreted as vectors. Instead, graphs are the better representations. In the remainder of this paper, we are going to shift our attention to graphs (inputs) instead of vectors in $\mathbb{R}^d$.

2 Graph Kernels and Quantum Random Walks

For graph-structured data, graph kernels are designed measure the similarity between graphs. Now, we have two basic questions: **How similar are two vertices of a graph? How similar are two graphs?** We commence by reviewing the classical graphs kernels and then quantum random walks.

2.1 Diffusion kernel of a pair of vertices

Kondor and Lafferty proposed the diffusion kernel on a graph [KL02] that answered the question how to measure the similarity between two vertices on a graph. It was then generalized to regularization case [KSB09].

Definition 6 (Adjacency matrix)**.** Given a (undirected, unweighted) graph $G = (V, E)$, its *adjacency matrix* $\hat{A}$ is defined as

$$\hat{A}_{v,v'} := \begin{cases} 1, & (v, v') \in E \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where the matrix entry is 1 if the two vertices v, v' (labels of the column and the row) are connected by an edge, otherwise the entry is 0.

Definition 7 (Graph Laplacian)**.** With the adjacency matrix $\hat{A}$, the graph Laplacian is obtained by

$$\hat{\mathfrak{L}} := \hat{A} - \hat{D} \tag{6}$$

where $\hat{D}_{v,v} := \deg(v) = \sum_v [\hat{A}]_{v,v'}$ is the diagonal degree matrix.

Remark 2. The adjacency matrix and graph Laplacian of a weighted graph can be defined in the similar manner. Graph Laplacian $\hat{\mathfrak{L}}$ is the discrete version of (continuous) Laplacian operator ∇^2 [Chu97]. The matrix exponential of any? $\hat{H}$, i.e. $e^{\beta \hat{H}}$, is a valid kernel (p.s.d). [to verify for $e^{-\mathrm{i}t\hat{H}}$]

Definition 8 (Diffusion kernel)**.** The *diffusion kernel* of a graph is defined as the exponential of the Laplacian

$$\mathcal{K}(v, v') := \left[\sum_k \frac{\beta^k}{k!} \hat{\mathfrak{L}}^k \right]_{v,v'} = \left[e^{\beta \hat{\mathfrak{L}}} \right]_{v,v'} \tag{7}$$

where the input space $\mathcal{X}$ is the set of vertices V of graph G.

2.1.1 Diffusion (heat) equation, and random walks

The continuous-time random walk on a graph is defined as the solution of the differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t} p_j(t) = \sum_{k \in V} \hat{\mathfrak{L}}_{jk}\ p_k(t), \tag{8}$$

where $p_j(t)$ denotes the probability associated with vertex j at time t and $\hat{\mathfrak{L}}$ is the Graph Laplacian. The simple *random walk* (discrete-space, discrete-time)

$$p_j(t+1) = \frac{p_{j+1}(t) + p_{j-1}(t)}{2} \tag{9}$$

is the finite difference form of *heat equation*

$$\frac{\partial p(t,x)}{\partial t} = \frac{\partial^2 p(t,x)}{\partial x^2} \tag{10}$$

which is continuous-space, continous-time case. The solution to Eq. (10) is called *heat kernel* $\mathcal{K}(t;x,x') = \frac{1}{4\pi t} e^{-|x-x'|^2/4t}$ (Gaussian).

2.2 Continuous-time quantum random walk

The continuous-time quantum random walk [CFG02] is the quantum analogue of classical diffusion (continuous-time random walk). By a direct observation, Eq. (8) is very similar to the time-dependent (evolution) schrodinger equation governed by a Hamiltonian operator $\hat{H}$

$$\mathrm{i}\hbar \frac{\mathrm{d}}{\mathrm{d}t} |\psi\rangle = \hat{H} |\psi\rangle \tag{11}$$

except that the factor of $\mathrm{i}\hbar$.

Definition 9 (Quantum propagator)**.** *Quantum propagator* is a function that specifies the probability amplitude for a quantum particle to travel from one place to another in a given period of time. This propagator may also be written as the *transition amplitude*.

$$\mathcal{K}(x,t;x',t') = \left\langle x \middle| e^{-\mathrm{i}(t'-t)\hat{H}} \middle| x' \right\rangle = \left\langle x \middle| \hat{U} \middle| x' \right\rangle \tag{12}$$

Interestingly, this quantity is also called *quantum kernel* (much earlier than the concept of kernel tricks in machine learning). The propagator of a one-dimensional free particle can be evaluated

$$\mathcal{K}_{free}(x_I,0;x_F,t) = \langle x_F | e^{-\mathrm{i}t\hat{p}^2/(2m\hbar)} | x_I \rangle = \sqrt{\frac{m}{2\pi \mathrm{i}\hbar t}} \exp\left(\frac{\mathrm{i}}{\hbar}\frac{m(x_F-x_I)^2}{2t}\right). \tag{13}$$

by *path integral (Lagrangian) formalism* (see Appendix B.1). Eq. (13) is a Gaussian form again.

2.3 Graph kernels of a pair of graphs

The graph kernel of a pair of graphs was firstly proposed by Haussler [Hau99], called R-convolution kernel. Graphs kernels are designed to compare the similarity of each of the decompositions of a pair of graphs. Different kernels are defined, depending on how the graphs are decomposed. Most R-convolution kernels count the number of isomorphic substructures in the two graphs. random walk kernel (shortest paths) [Vis+10]. survey [KJM20]

Definition 10 (Products of graphs)**.** The *direct product of graphs*, $G_\times(V_\times, E_\times) := G(V,E) \times G'(V',E')$

$$V_\times := \left\{(v,v') \in V \times V'\right\}, \quad E_\times := \left\{\left((v_i,v_i'),(v_j,v_j')\right) \subseteq V_\times \times V_\times : (v_i,v_j) \in E \wedge (v_i',v_j') \in E'\right\} \tag{14}$$

where $V \times V'$ is the Cartesian product of two sets.

Remark 3. The adjacency matrix of the product graph $\hat{A}_\times = \hat{A} \otimes \hat{A}'$ and $\exp\left(\hat{A}_\times\right) = \exp\left(\hat{A}\right) \otimes \exp\left(\hat{A}'\right)$. Performing a random walk on the direct product graph is equivalent to performing a simultaneous random walk on G and G'. However, in general, the Laplacian of the direct product graph $\hat{\mathfrak{L}}_\times \neq \hat{\mathfrak{L}} \otimes \hat{\mathfrak{L}}'$.

Definition 11 (Graph kernel)**.** Given a pair of graphs (G, G'), a *graph kernel* is a function (mapping) $\mathcal{K}(G,G') : \{G\} \times \{G\} \to \mathbb{R}$

$$\mathcal{K}(G,G') = \frac{1}{|G||G'|} \sum_k \frac{\lambda^k}{k!} \mathbf{e}^\intercal A_\times \mathbf{e} = \frac{1}{|G||G'|} \mathbf{e}^\intercal \exp\left(\beta \hat{A}_\times\right) \mathbf{e} \tag{15}$$

where $\mathbf{e}$ is ...

Graph kernels based on random walks count the number of (label sequences along) walks that two graphs have in common.

Definition 12 (Random walk kernel)**.**

$$\mathcal{K}_\times(G, G) = ... \tag{16}$$

Definition 13 (Factor graph)**.** *factor graph ...*

For diffusion processes on factor graphs, the kernel is given by the product of kernels on the constituents, that is

$$\mathcal{K}((i, i'), (j, j')) = \mathcal{K}(i, j)\mathcal{K}'(i', j'). \tag{17}$$

Remark 4. A natural question to ask is the following: since diffusion can be viewed as a **continuous time** limit of random walks, can the ideas behind the random walk kernel be extended to diffusion? Unfortunately, the Laplacian of the product graph does not decompose into the Kronecker product of the Laplacian matrices of the constituent graphs; this rules out a straightforward extension. Is it feasible to extend by discrete-time quantum random walk (need coin space) [AKR05] or Szegedy's walk formalism [Sze04].

2.4 Discrete-time quantum random walk

Discrete-time (DT) quantum random walk is the quantum analogy of classical DT random walk c.f. Eq. (9). The significant difference between CT and DT quantum random walk is that DT quantum random walk requires a coin space (flip a quantum coin at each move) to ensure the unitarity of the quantum walk [Amb+01]. Consequently, they have different behaviors. [TODO]

2.5 Relations, difference, and examples

2.5.1 Chain

Consider a 1-dimensional lattice (chain/line) of N vertices, the classical diffusion kernel between two positions $z_I, z_F \in [N]$

$$\mathcal{K}_{\mathrm{c}}(z_F, z_I) = \frac{1}{N}\sum_{\nu=1}^{N} e^{-2\beta(1-\cos\frac{2\pi\nu}{N})}\cos\left(\frac{2\pi\nu(z_I - z_F)}{N}\right). \tag{18}$$

Quantum propagator (CT quantum random walk)

$$\begin{aligned}\mathcal{K}_{\mathrm{q}}(z_F, z_I) = \langle z_F|e^{-\mathrm{i}t\hat{H}_0}|z_I\rangle &= \sum_{\nu=1}^{N} e^{-\mathrm{i}2t\cos(\frac{2\pi}{N}\nu)}\exp\left(\mathrm{i}\frac{2\pi\nu(z_I - z_F)}{N}\right) \\ &\approx e^{2\mathrm{i}t}(-\mathrm{i})^d J_d(2t) \qquad \text{(large N approx)}\end{aligned} \tag{19}$$

where $J(\cdot)$ is the Bessel function and d is the distance between the initial and final positions z_I, z_F.

Remark 5. The random walk on this graph starting from the origin (in either continuous or discrete time) typically moves a distance proportional to $\sqrt{t}$ in time t. In contrast, quantum random walks spread linearly in time t. see Fig. 2

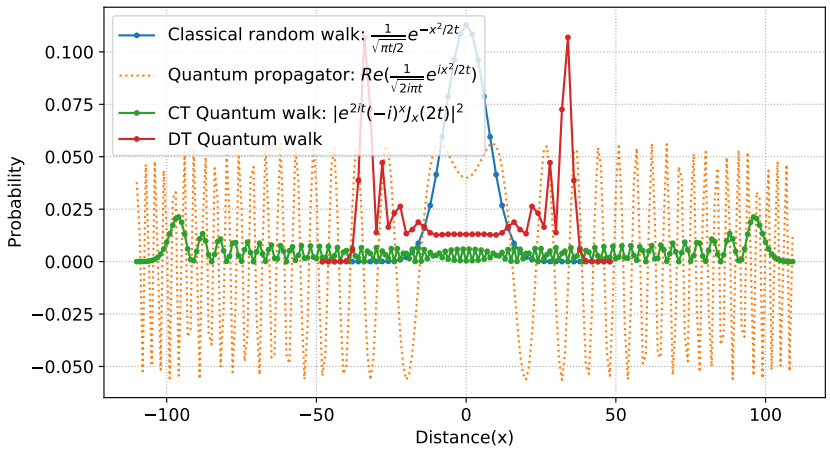

Figure 2: The probability distributions of different walks in 1d space after 50 steps.

2.5.2 Complete graph

the classical diffusion kernel: $\hat{H}_{ij} = 1 - \delta(i,j)N$

$$\mathcal{K}_{\mathrm{c}}(i,j) = \frac{1 - \hat{H}_{ij}e^{-N\beta}}{N} \tag{20}$$

CT quantum random walk

$$\mathcal{K}_{\mathrm{q}}(i,j) = \tag{21}$$

DT ...

2.5.3 Hyercube

The classical diffusion kernel on a n-dimensional hypercube

$$\mathcal{K}_{\mathrm{c}}(\mathbf{x},\mathbf{x}') \propto \left(\frac{1-e^{-2\beta}}{1+e^{-2\beta}}\right)^{d(\mathbf{x},\mathbf{x}')} = (\tanh\beta)^{d(\mathbf{x},\mathbf{x}')} \tag{22}$$

which only depends on the Hamming distance $d(\mathbf{x},\mathbf{x}')$ between two binary strings $\mathbf{x}$ and $\mathbf{x}'$. Quantumly, the unitary evolution operator of quantum (continuous-time) diffusion

$$e^{-\mathrm{i}t\hat{A}} = \prod_{j=1}^{n} e^{-\mathrm{i}t\hat{X}^{(j)}} = \bigotimes_{j=1}^{n} \begin{pmatrix} \cos t & -\mathrm{i}\sin t \\ -\mathrm{i}\sin t & \cos t \end{pmatrix}. \tag{23}$$

Discrete-time ... [MR01]

Remark 6. Consider the continuous or discrete time random walk starting from the vertex $\mathbf{x}$. The probability of reaching the opposite vertex $\bar{\mathbf{x}}$ is exponentially small at any time, since the walk rapidly reaches the uniform distribution (no stationary distribution?) over all 2^n vertices of the hypercube. So this simple example shows that random and quantum walks can exhibit radically different behaviors.

2.5.4 Tree

Kondor and Lafferty [KL02] discuss the diffusion kernel of the (infinite) 3-regular tree and binary rooted tree

$$\mathcal{K}_{\mathrm{c}}(i,j) = \frac{1}{2}\mathcal{K}_R^{(3)}(d(i,j)) + \frac{1}{2}\mathcal{K}_R^{(3)}(d(i,r)+d(r,j)+1) \tag{??}$$

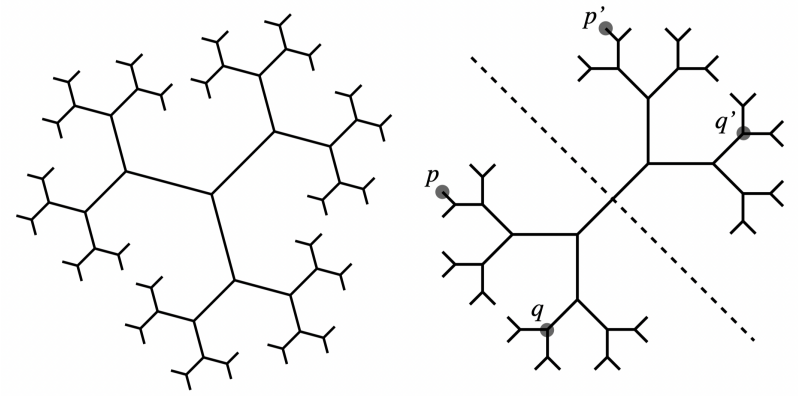

Figure 3: The three-regular tree (left), which extends to infinity in all directions. A little bending of the branches shows that it is isomorphic to two rooted binary trees joined at the root (right). [KL02]

where r designates the root and d measures distances on the binary tree. On the other hand, Childs et. al [Chi+03] investigate the CT quantum random walk on another kind of tree and show an exponential speedup over classical computation.

Problem 1 (Glued tree)**.** Consider a graph obtained by starting from two balanced binary trees of height n, and joining them by a random cycle of length $2 \cdot 2^n$ that alternates between the leaves of the two trees. (e.g. Fig. 4 when $n = 4$)

- **Input:** a such glued tree in its Adjacency matrix $\hat{A}$ representation, oracle? to entries of $\hat{A}$
- **Output (goal):** find the exit (start from entrance)

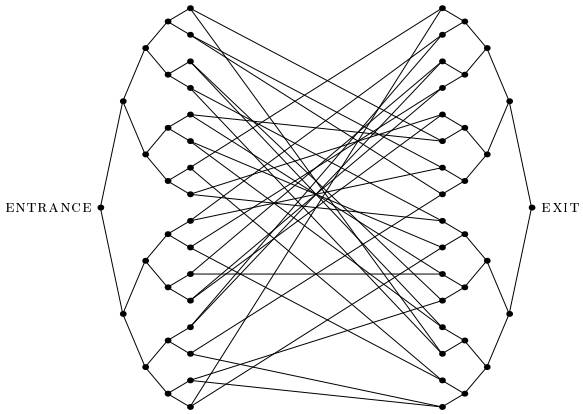

Figure 4: A typical graph exhibits exponential (query) speedup over classical model in graph matrix representation. [Chi+03]

2.5.5 Cayley graphs (TODO)

Definition 14 (Cayley graph)**.** Cayley graph is a graph that encodes the abstract structure of a group.

3 Quantum Advantages and Speedups

3.1 Related works

There have been several attempts to introduce kernel method to quantum machine learning.

3.1.1 Quantum SVM and kernel tricks

Quantum version of SVM was proposed [RML14] to exploit the power of quantum computer, where an exponential improvement can be achieved if data is provided in a coherent superposition. It is shown that a quantum support vector machine can be implemented with $O(\log md)$ run time in both training and classification stages. The performance in d arises due to a fast quantum evaluation of inner products. However, when data is provided in the conventional way, i.e. from a classical computer, then the methods cannot be applied [Tan19].

Definition 15 (Quantum feature map)**.** The *quantum feature map* is the direct quantum analogy of classical feature map.

$$\Phi(\mathbf{x}) : \Omega \to |\Phi(\mathbf{x})\rangle\langle\Phi(\mathbf{x})|\,, \tag{24}$$

where the quantum state is written in the density matrix representation. The quantum feature map $\Phi(\mathbf{x})$ is realized by applying a unitary quantum circuit $\hat{U}_{\Phi(\mathbf{x})}$ to a reference state $|\mathbf{0}\rangle$.

Definition 16 (Quantum kernel estimation)**.** Straightforwardly, the *quantum kernel estimation* is the *Hilbert-Schmidt* inner product between (feature) density matrices (c.f. Eq. (24))

$$\mathcal{K}(\mathbf{x},\mathbf{x}') = \mathrm{Tr}\left(|\Phi(\mathbf{x})\rangle\langle\Phi(\mathbf{x})| \cdot |\Phi(\mathbf{x}')\rangle\langle\Phi(\mathbf{x}')|\right) = \left|\langle\Phi(\mathbf{x})|\Phi(\mathbf{x}')\rangle\right|^2 = \left|\,\langle\mathbf{0}|\hat{U}^\dagger_{\Phi(\mathbf{x})}\hat{U}_{\Phi(\mathbf{x}')}|\mathbf{0}\rangle\right|^2 \tag{25}$$

This is exactly a Quantum propagator.

Naturally, the *quantum kernel estimation* [SK19] [Hav+19] has been studied to enhance the performance of quantum SVM. There are mainly two approaches.

3.1.2 Quantum variational classification (explicit method)

The first one is based on variational quantum circuit: generates a separating hyperplane in the quantum feature space. the quantum variational classifier builds on [Mit+18] [FN18] and operates through using a variational quantum circuit to classify a training set in direct analogy to conventional SVMs. To obtain an advantage over classical approaches, it is necessary to implement a map based on circuits that are hard to simulate classically.

1. a classical datum $\mathbf{x} \in \Omega$ is mapped to a quantum (feature) state by applying a unitary circuit $\hat{U}_{\Phi(\mathbf{x})}$ to a reference (initial) state $|0\rangle^n$
2. a **short depth quantum** circuit $W(\boldsymbol{\theta})$. iteratively optimize the circuit variational parameters $\boldsymbol{\theta}$ by minizing empirical error (cost function)
3. for binary classification, apply a **binary measurement** in Z-basis
4. to obtain the empirical distribution $p_y(\mathbf{x})$, perform repeated measurement shots. then assign the label according to p_y?

By the hardness assumption [BMS16], the circuit is hard to simulate classically. The hardness of estimating the kernel with classical resources is of course only a necessary and not always sufficient condition to obtain a quantum advantage.

3.1.3 Quantum kernel estimation (implicit method)

In the second method, quantum computer is used only to estimate the kernel function (matrix). And implement a conventional (classical) SVM to generate the separating hyperplane rather than using a variational quantum circuit.

The kernel entries are the fidelities between different feature vectors. The overlap can be estimated directly from the transition amplitude Quantum kernel estimation. Measuring the final state in the Z-basis R-times and record the number of $|0^n\rangle$. The frequency of this string is the estimate of the transition probability.

3.1.4 Quantum graph kernels

In [Bai+15], a quantum graph kernel is defined in terms of quantum Jensen Shannon as a metric of dissimilarity of graphs density matrices associated with the evolution of continuous-time quantum random walks on graphs. In classical information theory, the Jensen-Shannon divergence $\mathrm{JSD}(p, p')$ is a similarity measure between probability distributions. Analogously, the quantum version of Jensen-Shannon divergence is defined as the distance measure between mixed quantum states (density matrices).

Definition 17. Given a graph $G(V, E)$, the von Neumann entropy of G is defined as

$$H_N(\rho_G) = -\operatorname{Tr}(\rho_G \log \rho_G) = -\sum_j^{|V|} \lambda_j \log \lambda_j \tag{26}$$

where ρ_G is ..random walk?? and λ is .... Then, the *quantum Jensen-Shannon divergence* of two density operators ρ and ρ' is defined as

$$\mathrm{QJSD}(\rho, \rho') = H_N\left(\frac{\rho + \rho'}{2} - \frac{1}{2}H_N(\rho)\right) - \frac{1}{2}H_N(\rho') \tag{27}$$

Remark 7. p.s.d

3.1.5 Quantum diffusion map

Inspired by random walk on graphs, *diffusion map* (DM) is a class of **unsupervised** machine learning that offers automatic identification of **low-dimensional data structure hidden in a high dimensional dataset**. Though both of SVM and DM are machine learning techniques that utilize kernel tricks, DM is designed for dimension reduction, the 'reverse direction' of the kernel trick in SVM. Like other dimension reduction technique, DM entails the calculate the eigenvalue/vectors (SVD). Thus, classical dimensionality reduction can be computationally prohibitive for a large data sample. However, under moderate assumptions of accessibility to certain features of full-scale quantum computers, matrix exponentiation-based quantum algorithms have been proposed to perform SVD more efficiently [ref]. The quantum diffusion map (qDM) consists of 5 major steps:

1. Coherent state data encoding scheme,
2. a natural construction of kernel matrix from coherent states, Eq. (28)
3. a scheme to construct the Markov transition matrix from the kernel matrix,
4. the eigen-decomposition of the transition matrix (quantumly $\mathcal{O}(\mathrm{poly}\log N)$?); classically $\mathcal{O}(N^3)$
5. and extracting relevant quantum information to construct diffusion map classically. and classically constructs the diffusion map via the readout (tomography) of the eigenvectors, giving a total expected runtime proportional to $N^2 \,\mathrm{poly}\log N$.

Rather than using mere Euclidean distance as a similarity measure between data points, manifold learning approach assigns the connectivity among data points in their neighborhood as a proxy for proximity between points. In DM, the similarity matrix between a pair of data vectors, or equivalently the weighted edge between a pair of graph vertices, is often taken to be a Gaussian kernel Eq. (4):

$$W_{ij} := \mathcal{K}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\frac{\left\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right\|_2^2}{2\sigma}\right) \tag{28}$$

where the adjustable parameter σ, called the bandwidth, sets the scale of neighborhood. Given a graph with weighted edges W_{ij}, DM assigns a *discrete-time random walk* on a data-induced graph, where the Markov transition probability from vertex i to j is given by the normalized weighted edge,

$$P_{ij} = \frac{W_{ij}}{\sum_j W_{ij}}, \quad \tilde{P} = D^{-1}W. \tag{29}$$

Definition 18 (Coherent state)**.** A *coherent state* is ...

$$|\alpha\rangle = e^{-\frac{1}{2}|\alpha|^2} \sum_{n=0}^{\infty} \frac{\alpha^n}{\sqrt{n!}} |n\rangle = e^{-\frac{1}{2}|\alpha|^2} e^{\alpha \hat{a}^\dagger} |0\rangle \tag{30}$$

where $|n\rangle$ basis and $\hat{a}^\dagger$ is the creation operator [ref].

Consider two n tensor product of (canonical) coherent states

$$|\boldsymbol{\alpha}\rangle = |\alpha_1\rangle \otimes |\alpha_2\rangle \cdots \otimes |\alpha_n\rangle\,, \quad \left|\boldsymbol{\alpha}'\right\rangle = \left|\alpha_1'\right\rangle \otimes \left|\alpha_2'\right\rangle \cdots \otimes \left|\alpha_n'\right\rangle \tag{31}$$

with $\mathbf{x} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ and $\mathbf{x}' = (\alpha_1', \alpha_2', \ldots, \alpha_n')$, the kernel reads [CY17]

$$\left|\langle \boldsymbol{\alpha} | \boldsymbol{\alpha}' \rangle\right|^2 = \exp\left(-\left\|\mathbf{x} - \mathbf{x}'\right\|^2\right) \tag{32}$$

which is the Gaussian kernel c.f. Eq. (4) and Eq. (28).

3.2 Provable quantum speedups

The core challenge to provable quantum speedup and substantial improvement for practical problems

3.2.1 Quantum linear algebra toolbox (subroutines)

Firstly, provable speedups by quantum linear algebra algorithms:

- HHL (quantum) algorithm for solving linear equation systems [HHL09], qSVD (eigen decomposition) [Gil+19]; quantum linear algebra (qMAT) [Zha+19]: matrices multiplication, matrix inversion, matrix exponentiation,
- quantum (evolution) simulation techniques: product formula, LCU, etc.
- quantum Fourier transform (QFT), quantum phase estimation.
- FFT by quantum algorithms: It is shown that quantum circuits are a natural choice for Fourier space neural architectures affording a super-exponential speedup in computing the matrix elements of $\mathbb{S}_n$-Fourier coefficients compared to the best known classical (FFT) over the symmetric group. [Zhe+22] [KSB09]

3.2.2 Quantum Fourier Transform (QFT) and algebraic problems

hidden subgroup problem solved by quantum Fourier transformation[CvD10]; rigorous and robust quantum speedup with Discret Logarithm problem [LAT21] [Gli+21]

Problem 2 (Hidden subgroup problem)**.** Given a group $\mathbb{G}$ and a black-box function $f : \mathbb{G} \to S$, f is promised to satisfy

$$f(x) = f(y) \text{ iff } x^{-1}y \in \mathbb{H} \tag{33}$$

for some unknow subgroup $\mathbb{H} \leq \mathbb{G}$. We say such a function f hides a subgroup $\mathbb{H}$. The goal of *hidden subgroup problem* (HSP) is to learn $\mathbb{H}$ (specified in terms of a generating set) using queries to f.

Remark 8. Simon's problem [Sim97] is a HSP with $\mathbb{G} = \mathbb{Z}_2^n$ and $\mathbb{H} = \{0, s\}$ for some $s \in \mathbb{Z}_2^n$.

Problem 3 (Discrete Logarithm problem)**.** The *discrete logarithm problem* (DLog) is defined as

- **Input:** a prime p, a primitive element (generator) g of $\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\}$, and an element $y = g^x \pmod p \in \mathbb{Z}_p^*$ with some unknow x. black-box function f?
- **Output (goal):** find $x = \log_g y$

DLog is a special case of HSP with ... subgroup

reducible to decision version $\mathsf{DLog}_{1/2}$, then classification version.

$$\mathsf{DLog} \leq \mathsf{DLog}_{1/2} \leq \mathsf{DLog} \tag{34}$$

$\leq$ represents polynomial (time) reduction.

covariant quantum kernels for the data with group structures [Gli+21]. Exploiting group structure and learning problems for group data have already been investigated for classical kernels [Kon08]. The fiducial state $|\phi\rangle$ is chosen as some reference state from the representing Hilbert space. In principle, any state that can be prepared efficiently may be chosen. However, the choice will greatly affect the performance of the classifier as we will discuss shortly.

Definition 19 (Covariant)**.** *covariant* feature map circuits that we call covariant feature maps.

Problem 4 (labeling cosets with error)**.** *labeling cosets with error*

- **Input:** given a group $\mathbb{G}$ and subgroup $\mathbb{H}$, we can define two left-cosets $\mathbb{C}_\pm = c_\pm\mathbb{H}$ by choosing c_+, c_- randomly ...
- **Output: ?**

3.2.3 Permutation, symmetry, and graph properties

It is well-known that structure of a problem is necessary for quantum speedup [AA14]. only polynomial speedup for total problem [Bea+01] [Aar+20]. An illustrative example is that quadratic speedup is optimal for unstructured (Grover) search [Gro97], while exponential speedup is widely believed for integer (prime) factorization (promise/partial problem) [Sho97]. Moreover, fully symmetric (partial) functions do not admit exponential quantum speedup in the query model. [Ben+20]. Graph property test ...

$$\leq \tag{35}$$

reduction to (modified) glued tree Fig. 4

3.3 Heuristic quantum advantages for learning physics systems

Previous works evaluate the performance on the standard graph datasets from both bioinformatics and computer vision. Instead, we could exploit the structures inherent in physics systems and make use of quantum advantages.

3.3.1 Classical machine learning for quantum physics problem

Aligned with the development of quantum machine learing, classical machine learning is being applied to tackle quantum many-body physics (determining phase (transition)) and particle physics. It is shown that a standard feed-forward neural network can be trained to detect multiple types of order parameter directly from raw state configurations sampled with Monte Carlo [CM17]. Moreover, the application of such techniques to problems of greater interest in modern condensed matter, such as disordered or topological phases, where no conventional order parameter exists. [CT17]

Remark 9. A straightforward implementation of supervised training fails to classify a test set containing samples of the two states to an accuracy over 50% - equivalent to simply guessing. Such failures typically occur because the neural network overfits to the training set. To overcome this difficulty we consider a convolutional neural network (CNN) [ref] which readily takes advantage of the two-dimensional structure of the input configurations, as well as the **translational invariance** of the model.

3.3.2 Groups and symmetries in physics and machine learning

"Group Theoretical Methods in Machine Learning" [Kon08]; "Lorentz Group Equivariant Neural Network for Particle Physics" [Bog+20]; *Symmetry Group Equivariant Architectures for Physics* [Bog+22];

One of the key properties of classical CNNs is equivariance, which roughly states that if the input to the neural network is shifted, then its activations translate accordingly.

Definition 20 (Equivariance)**.** Given a group $\mathbb{G}$ and the actions $\rho : \mathbb{G} \times X \to X$ and $\rho' : \mathbb{G} \times Y \to Y$, a map $f : X \to Y$ is said to be *equivariant* if

$$\forall x \in X, g \in \mathbb{G}, f(\rho(g, x)) = \rho'(g, f(x)) \tag{36}$$

Remark 10. In particular, it has been recognized that by constructing neural networks that operate on the basis of irreducible representations of the group (so-called Fourier space neural networks), and group equivariant convolution is easy to implement because it simply reduces to matrix multiplication. The major difficulty is that the key ingredient of the success of CNNs - translation invariance - lacks a mathematically rigorous quantum counterpart due to the discrete spectrum of spin-based quantum circuits. the natural form of equivariance in quantum circuits is permutation equivariance. [Zhe+22]!?

4 Experiments

4.1 Datasets and benchmark

Firstly, we will validate our method (by preliminary experiment) on artificial data (of group structures). Then, we test the performance for prototypical machine learning tasks and physics datasets.

4.1.1 Artificial data

We generate artificial data that can be fully separated by our feature map.

4.1.2 Real-world dataset

conventional graph datasets

- UCI [KL02],
- protein [ref],
- The MUTAG dataset consists of graphs representing chemical compounds labeled ..;

physics datasets

- particle physics: JET tagging task [Bog+20];
- quantum many-body physics: determine phase transition [CM17], topological order [CCL19]

5 Discussion and Conclusion

- formalize quantum graph kernels (algorithm) with quantum random walk, provable speedup
- quantum graph kernel for the data with group structures
- quantum machine learning for (practical dataset) physics problem (symmetries)
- whether low-depth circuit for NISQ. analog computing?

References

[AA14] Scott Aaronson and Andris Ambainis. "The Need for Structure in Quantum Speedups". Feb. 6, 2014. arXiv: 0911.0996 [quant-ph] (cit. on p. 12).

[Aar+20] Scott Aaronson et al. "Quantum Implications of Huang's Sensitivity Theorem". Apr. 27, 2020. arXiv: 2004.13231 [quant-ph] (cit. on p. 12).

[AKR05] Andris Ambainis, Julia Kempe, and Alexander Rivosh. Coins Make Quantum Walks Faster. Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms. SODA '05. USA: Society for Industrial and Applied Mathematics, Jan. 23, 2005, pp. 1099–1108. arXiv: quant-ph/0402107 (cit. on p. 6).

[Amb+01] Andris Ambainis et al. One-Dimensional Quantum Walks. Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing - STOC '01. The Thirty-Third Annual ACM Symposium. Hersonissos, Greece: ACM Press, 2001, pp. 37–49 (cit. on p. 6).

[Bai+15] Lu Bai et al. A Quantum Jensen–Shannon Graph Kernel for Unattributed Graphs. *Pattern Recognition* 48.2 (Feb. 2015), pp. 344–355 (cit. on p. 10).

[Bea+01] Robert Beals et al. Quantum Lower Bounds by Polynomials. *J. ACM* 48.4 (July 1, 2001), pp. 778–797. arXiv: quant-ph/9802049 (cit. on p. 12).

[Ben+20] Shalev Ben-David et al. Symmetries, Graph Properties, and Quantum Speedups. *2020 IEEE 61st Annu. Symp. Found. Comput. Sci. FOCS* (Nov. 2020), pp. 649–660. arXiv: 2006.12760 (cit. on pp. 2, 12).

[Bia+17] Jacob Biamonte et al. Quantum Machine Learning. *Nature* 549.7671 (Sept. 2017), pp. 195–202. arXiv: 1611.09347 (cit. on p. 17).

[BMS16] Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. Average-Case Complexity versus Approximate Simulation of Commuting Quantum Computations. *Phys. Rev. Lett.* 117.8 (Aug. 18, 2016), p. 080501. arXiv: 1504.07999 (cit. on p. 9).

[Bog+20] Alexander Bogatskiy et al. Lorentz Group Equivariant Neural Network for Particle Physics. Proceedings of the 37th International Conference on Machine Learning. International Conference on Machine Learning. PMLR, Nov. 21, 2020, pp. 992–1002 (cit. on pp. 13, 14).

[Bog+22] Alexander Bogatskiy et al. Symmetry Group Equivariant Architectures for Physics. Mar. 11, 2022. arXiv: 2203.06153 [astro-ph, physics:hep-ex, physics:hep-ph] (cit. on p. 13).

[CCL19] Iris Cong, Soonwon Choi, and Mikhail D. Lukin. Quantum Convolutional Neural Networks. *Nat. Phys.* 15.12 (Dec. 2019), pp. 1273–1278. arXiv: 1810.03787 [cond-mat, physics:quant-ph] (cit. on pp. 2, 14, 17).

[CFG02] Andrew M. Childs, Edward Farhi, and Sam Gutmann. An Example of the Difference between Quantum and Classical Random Walks. *Quantum Inf. Process.* 1.1/2 (2002), pp. 35–43. arXiv: quant-ph/0103020 (cit. on p. 5).

[Chi+03] Andrew M. Childs et al. Exponential Algorithmic Speedup by Quantum Walk. *Proc. Thirty-Fifth ACM Symp. Theory Comput. - STOC 03* (2003), p. 59. arXiv: quant-ph/0209131 (cit. on p. 8).

[Chu97] Fan R. K. Chung. Spectral Graph Theory. Regional Conference Series in Mathematics no. 92. Providence, R.I: Published for the Conference Board of the mathematical sciences by the American Mathematical Society, 1997. 207 pp. (cit. on p. 4).

[CM17] Juan Carrasquilla and Roger G. Melko. Machine Learning Phases of Matter. *Nature Phys* 13.5 (5 May 2017), pp. 431–434. arXiv: 1605.01735 (cit. on pp. 13, 14).

[CT17] Giuseppe Carleo and Matthias Troyer. Solving the Quantum Many-Body Problem with Artificial Neural Networks. *Science* 355.6325 (Feb. 10, 2017), pp. 602–606. arXiv: 1606.02318 (cit. on p. 13).

[CV95] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Mach Learn* 20.3 (Sept. 1995), pp. 273–297 (cit. on p. 2).

[CvD10] Andrew M. Childs and Wim van Dam. Quantum Algorithms for Algebraic Problems. *Rev. Mod. Phys.* 82.1 (Jan. 15, 2010), pp. 1–52. arXiv: 0812.0380 (cit. on p. 12).

[CY17] Rupak Chatterjee and Ting Yu. Generalized Coherent States, Reproducing Kernels, and Quantum Support Vector Machines. Feb. 2, 2017. arXiv: 1612.03713 [quant-ph] (cit. on p. 11).

[Dir45] P. A. M. Dirac. On the Analogy Between Classical and Quantum Mechanics. *Rev. Mod. Phys.* 17.2-3 (Apr. 1, 1945), pp. 195–199 (cit. on p. 18).

[FHS10] Richard P. Feynman, Albert R. Hibbs, and Daniel F. Styer. Quantum Mechanics and Path Integrals. Emended ed. Mineola, N.Y: Dover Publications, 2010. 371 pp. (cit. on p. 18).

[FN18] Edward Farhi and Hartmut Neven. Classification with Quantum Neural Networks on Near Term Processors. Aug. 30, 2018. arXiv: 1802.06002 [quant-ph] (cit. on p. 9).

[Gil+19] András Gilyén et al. Quantum Singular Value Transformation and beyond: Exponential Improvements for Quantum Matrix Arithmetics. *Proc. 51st Annu. ACM SIGACT Symp. Theory Comput.* (June 23, 2019), pp. 193–204. arXiv: 1806.01838 (cit. on p. 11).

[Gli+21] Jennifer R. Glick et al. "Covariant Quantum Kernels for Data with Group Structure". May 7, 2021. arXiv: 2105.03406 [quant-ph] (cit. on pp. 2, 12).

[Gro97] Lov K. Grover. Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Phys. Rev. Lett.* 79.2 (July 14, 1997), pp. 325–328 (cit. on p. 12).

[Hau99] David Haussler. Convolution Kernels on Discrete Structures. *UCSC-CRL-99-10* (1999) (cit. on p. 5).

[Hav+19] Vojtech Havlicek et al. Supervised Learning with Quantum Enhanced Feature Spaces. *Nature* 567.7747 (Mar. 2019), pp. 209–212. arXiv: 1804.11326 (cit. on pp. 2, 9).

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum Algorithm for Solving Linear Systems of Equations. *Phys. Rev. Lett.* 103.15 (Oct. 7, 2009), p. 150502. arXiv: 0811.3171 [quant-ph] (cit. on p. 11).

[KJM20] Nils M. Kriege, Fredrik D. Johansson, and Christopher Morris. A Survey on Graph Kernels. *Appl Netw Sci* 5.1 (Dec. 2020), p. 6. arXiv: 1903.11835 [cs, stat] (cit. on p. 5).

[KL02] Risi Imre Kondor and John Lafferty. Diffusion Kernels on Graphs and Other Discrete Structures (2002), p. 8 (cit. on pp. 4, 7, 8, 14).

[Kog79] John B. Kogut. An Introduction to Lattice Gauge Theory and Spin Systems. *Rev. Mod. Phys.* 51.4 (Oct. 1, 1979), pp. 659–713 (cit. on p. 18).

[Kon08] Risi Kondor. "Group Theoretical Methods in Machine Learning". Thesis. Columbia University, 2008 (cit. on pp. 2, 12, 13).

[KSB09] Risi Kondor, Nino Shervashidze, and Karsten M. Borgwardt. The Graphlet Spectrum. Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09. The 26th Annual International Conference. Montreal, Quebec, Canada: ACM Press, 2009, pp. 1–8 (cit. on pp. 4, 11).

[LAT21] Yunchao Liu, Srinivasan Arunachalam, and Kristan Temme. A Rigorous and Robust Quantum Speed-up in Supervised Machine Learning. *Nat. Phys.* 17.9 (Sept. 2021), pp. 1013–1017. arXiv: 2010.02174 [quant-ph] (cit. on pp. 2, 12).

[LMR14] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum Principal Component Analysis. *Nature Phys* 10.9 (Sept. 2014), pp. 631–633. arXiv: 1307.0401 (cit. on pp. 2, 17).

[Mit+18] Kosuke Mitarai et al. Quantum Circuit Learning. *Phys. Rev. A* 98.3 (Sept. 10, 2018), p. 032309. arXiv: 1803.00745 [quant-ph] (cit. on p. 9).

[MR01] Cristopher Moore and Alexander Russell. "Quantum Walks on the Hypercube". Apr. 28, 2001. arXiv: quant-ph/0104137 (cit. on p. 7).

[RML14] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum Support Vector Machine for Big Data Classification. *Phys. Rev. Lett.* 113.13 (Sept. 25, 2014), p. 130503. arXiv: 1307.0471 [quant-ph] (cit. on pp. 2, 9, 17).

[Sho97] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM J. Comput.* 26.5 (Oct. 1997), pp. 1484–1509. arXiv: quant-ph/9508027 (cit. on p. 12).

[Sim97] Daniel R. Simon. On the Power of Quantum Computation. *SIAM J. Comput.* 26.5 (Oct. 1, 1997), pp. 1474–1483 (cit. on p. 12).

[SK19] Maria Schuld and Nathan Killoran. Quantum Machine Learning in Feature Hilbert Spaces. *Phys. Rev. Lett.* 122.4 (Feb. 1, 2019), p. 040504. arXiv: 1803.07128 [quant-ph] (cit. on pp. 2, 9).

[Sze04] Mario Szegedy. "Spectra of Quantized Walks and a $\sqrt{\delta\epsilon}$ Rule". Jan. 9, 2004. arXiv: quant-ph/0401053 (cit. on p. 6).

[Tan19] Ewin Tang. A Quantum-Inspired Classical Algorithm for Recommendation Systems. *Proc. 51st Annu. ACM SIGACT Symp. Theory Comput.* (June 23, 2019), pp. 217–228. arXiv: 1807.04271 (cit. on p. 9).

[Tan21] Ewin Tang. Quantum Principal Component Analysis Only Achieves an Exponential Speedup Because of Its State Preparation Assumptions. *Phys. Rev. Lett.* 127.6 (Aug. 4, 2021), p. 060503. arXiv: 1811.00414 [quant-ph] (cit. on pp. 2, 17).

[Vis+10] S.V.N. Vishwanathan et al. Graph Kernels. *J. Mach. Learn. Res.* 11.40 (2010), pp. 1201–1242. arXiv: 0807.0093 (cit. on p. 5).

[Xu21] Jue Xu. "On Lagrangian Formalism of Quantum Computation". Dec. 7, 2021. arXiv: 2112.04892 [quant-ph] (cit. on p. 18).

[Zha+19] Liming Zhao et al. Compiling Basic Linear Algebra Subroutines for Quantum Computers. Feb. 27, 2019. arXiv: 1902.10394 [quant-ph] (cit. on p. 11).

[Zhe+22] Han Zheng et al. "Speeding up Learning Quantum States through Group Equivariant Convolutional Quantum Ans\"atze". Jan. 20, 2022. arXiv: 2112.07611 [math-ph, physics:quant-ph, stat] (cit. on pp. 11, 13).

A Machine Learning and Group Theory [TODO]

- supervised learning (classification, regression): SVM, neural network. (methods: gradient descent, backpropagation)
- unsupervised learning (clustering, dimension reduction): k-means, PCA
- reinforced learning not covered in this paper.

A.1 Machine learning

A.1.1 Details of SVM and kernel tricks [TODO]

A.1.2 Quantum machine learning

overview [Bia+17]

- supervised learning: quantum SVM [RML14], quantum neural network [CCL19];
- unsupervised learning: quantum PCA [LMR14] [Tan21].

A.2 Group theory and symmetries

permutation (symmetric) group $\mathbb{S}_n$;

A.2.1 Representation theory

group $\mathbb{G}$

B Symmetries in physics

B.1 Lagrangian formalism

In optics, Fermat's principle states that the path taken by a ray between two given points is the path that can be traveled in the least (extremum) time. A similar argument, *principle of least action*, was developed in classical mechanics:

Axiom 1 (Principle of least action)**.** *The actual path $q(t)$ taken by a classical system is the path that yields an extremum of its action $\mathcal{S}$. So, this principle is also called principle of stationary action. The action of the dynamics is the integral of Lagrangian over time*

$$\mathcal{S}[q(t)] := \int_{t_I}^{t_F} \mathcal{L}(q(t), \dot{q}(t); t)\, \mathrm{d}t \tag{37}$$

where $\mathcal{L}(q, \dot{q})$ is the Lagrangian in terms of generalized coordinate q and velocity $\dot{q}$ at certain time t.

The notion $\mathcal{S}[\cdot]$ reminds that action is a functional that takes a function (path) $q(t)$ as input. By varying the action, one have the equation of motion called *Euler-Lagrange equation*. This Lagrangian formalism was extended by Dirac [Dir45] and Feynman [FHS10] to explain quantum mechanics.

Axiom 2 (Path integral)**.** *The amplitude (probability) of a quantum system evolving from $|q_I\rangle$ to $|q_F\rangle$ in a time interval can be evaluated by (functional) integrating over all possible paths with fixed initial and final position*

$$\langle q_F | e^{-\mathrm{i}t\hat{H}/\hbar} | q_I \rangle = \int_{q(t_I)=q_I}^{q(t_F)=q_I} \mathcal{D}q\; e^{\mathrm{i}\mathcal{S}[q]/\hbar} \tag{38}$$

where the action defined in classical mechanics as Eq. (37).

The Larangian (path integral) formalism of quantum mechanics is proved to be equivalent to the well-known Schrödinger equation Eq. (11) [FHS10, Chp4] which is a differential equation determining the evolution of quantum state. In the classical limit (Planck's constant $\hbar \to 0$), Path integral reduces to Principle of least action because only the paths around the stationary point of the action contribute (the other paths' contributions frequently oscillate and cancel out).

B.1.1 Path integral and quantum computing

[Xu21]

B.2 Symmetries in physics with Lagrangians

B.2.1 Z

Ising model, $\mathbb{Z}_2$

B.2.2 U(1)

[Kog79] local, gauge symmetry

B.2.3 SU(2)

non-abelian, particle physics

B.2.4 SO(1,3)

Lorentz invariance