A Related Works

In this section, we provide background on quantum graph learning and graph neural networks that have the potential to be trained using quantum computing.

A.1 Graph Kernel Neural Network

Graph kernel neural networks [17, 10, 18] is a class of graph learning method combining the properties of both GNNs and GKs. Forward process of the model tends to transmit node information like GNNs [66], layer by layer, whereas the node (or graph) features live in the implicit reproducing kernel Hilbert space (RKHS) of a specific kernel [40]. The mainstream of graph kernel neural networks can be divided into completed and approximated approaches. For the former, the output is a kernel matrix where each entry denotes the similarity of graph pairs, afterwards support vector machin is used to perform classification or regression task. While the later generates the approximate feature of the finite projected RKHS at the expense of information loss.

We consider the completed approaches for the basis of our proposed quantum graph learning model, since the access to the explicit feature information requires measuring the relative quantum representations, which incurs quantum collapse [33]. In the next section, we demonstrate that the graph tangent kernel neural networks coincides with the condition of quantum parallel implementation by introducing the quantum aggregation transformation and the quantum kernel estimation techniques.

A.2 Quantum Graph Learning

Quantum graph learning aims at leveraging quantum physics to extract graph structural information, bringing up new possibilities for quantum computing applications. It is generally nontrivial to analyze classical data under the regime of quantum computing, since the encoding and decoding between classical vectors (or matrices) and their corresponding quantum states should be carefully designed. In addition, encoding the irregular graph data and diverse structure topology may incur different configurations of quantum models. Advanced contributions has developed some techniques to overcome these issues. A hybrid graph learning method developed by [70, 16] encode the structure information and generate a new adjacent matrix evaluating by the using quantum walk. The resulting adjacent information captures the global topological arrangement information for graph substructures. Adiabatic evolution [64] and conditional unitary [46] are applied to evolve the quantum systems dependent on the underlying graph structure. In addition, the node attribute is encoded using variational circuit [1] or a quantum random access memory [58]. Processing the encoded quantum representation of the original graph can be realized via either a naive quantum algorithm [64] or a hybrid method [11]. Then a post-processing operation is performed to further analyze the quantum output. A brief review about quantum graph learning is illustrated in Fig. 1. Generally speaking, the researchers exploit to encode the graph structure and node features in the quantum system through various schemes, and then process the information through quantum layers and auxiliary classical layers. Finally, the (quantum) results are decoded through post-processing. However, most quantum graph learning models requires that adjustable parameters in the quantum algorithm need to be updated frequently, where takes great computational overheads. Moreover, the classical components in post-processing may dominate the performance of the model, thus weakening the role of the quantum part. In this paper, We seek to establish a parameter-free quantum graph learning model to maximize the efficacy of quantum computing.

We notice that there are researches which are abbreviated as QNTK [59, 42], similar to ours nominally. But their definition is quite different from ours. The motivation of these two papers is to analyze the trainability and expressive power of variational quantum circuits through NTK. In contrast, in our work, QNTK is a metric measuring the similarity of two input graphs. In this context, NTK is the kernel that captures the dynamics of infinite-width GNNs, as well as the multi-head attention where the number of heads and the dimension of output go to infinity.

B More Analysis

B.1 Quantum Access Memory

Theorem 1 *Let* $|\mathbf{X}_p\rangle = \frac{1}{\|\mathbf{X}_p\|}\sum_{q=0}^{d-1}\mathbf{X}_{pq}|j\rangle$ *denotes the amplitude encoding of the* p*-th row of data* $\mathbf{X} \in \mathbb{R}^{n\times d}$*. There exists a data structure to store the entries of* $\mathbf{X}$ *into the QRAM which is stated as*

Table 3: Statistic information of the used datasets.

Dataset	MUTAG	PROTEINS	PTC	NCI1	IMDB-B	IMDB-M	ENZYMES	BZR	COX2
size	188	1113	344	4110	1000	1500	600	405	467
classes	2	2	2	2	2	3	6	2	2
attr. dim.	-	-	-	-	-	-	18	3	3
avg. nodes	18	39	26	30	20	13	32.6	35.8	41.2
avg. edges	20	73	51	32	97	66	62.1	38.3	43.5

i) $|p\rangle\,|0\rangle \rightarrow |p\rangle\,|\mathbf{X}_p\rangle$

ii) $|0\rangle \rightarrow \frac{1}{\|\mathbf{X}\|_F}\sum_p \|\mathbf{X}_p\|\,|p\rangle$

in time T for $p \in [n]$. Using the binary tree QRAM architecture proposed by [35], the time T to store and readout a new element scale logarithmically with respect to both n and d.

B.2 Inner Product Estimation

Theorem 2 *There exists a quantum operation $\mathcal{A}$ that evaluates the inner product of two quantum representations with respect to their d-dimensional classical vectors in time $O(\log d)$.*

Proof. By introducing an auxiliary register, with the initial state $|p\rangle|q\rangle\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)|0\rangle$, the map $\frac{1}{\sqrt{2}}(|p\rangle|q\rangle|0\rangle|0\rangle + |p\rangle|q\rangle|1\rangle|0\rangle) \rightarrow \frac{1}{\sqrt{2}}(|p\rangle|q\rangle|0\rangle|\,|\mathbf{X}_p\rangle\rangle + |p\rangle|q\rangle|1\rangle|\,|\mathbf{X}_q\rangle\rangle)$ can be performed in $O(\log d)$ for two quantum representations $|\mathbf{X}_p\rangle$ and $|\mathbf{X}_q\rangle$ with respect to their classical vectors $\mathbf{X}_p \in \mathbb{R}^d$ and $\mathbf{X}_q \in \mathbb{R}^d$. Applying a Hadamard gate on the third register, the state becomes

$$\frac{1}{2}|p\rangle|q\rangle\left(|0\rangle\left(|\mathbf{X}_p\rangle + |\mathbf{X}_q\rangle\right) + |1\rangle\left(|\mathbf{X}_p\rangle - |\mathbf{X}_q\rangle\right)\right). \tag{24}$$

The probability of measuring 0 on the third register is given by $P_{pq} = \frac{1+\langle\mathbf{X}_p|\mathbf{X}_q\rangle}{2}$. Thus the state defined by Eq. 24 can be reformulated as $|p\rangle|q\rangle\left(\sqrt{P_{pq}}\,|0, g_{pq}\rangle + \sqrt{1-P_{pq}}\,\middle|1, g'_{pq}\right\rangle\right)$ where $|g_{pq}\rangle$ and $\left|g'_{pq}\right\rangle$ are garbage states.

B.3 Amplitude Estimation

Theorem 3 *Given a unitary operator U such that $U : |0\rangle \mapsto \sqrt{p}|y\rangle|0\rangle + \sqrt{1-p}\,|y'\rangle\,|1\rangle$ in time T, where $p > 0$ is the probability of measuring 0, it is possible to obtain the state $|y\rangle|0\rangle$ using $O(\frac{T}{\sqrt{p}})$ queries to U, or to estimate p with relative error δ using $O(\frac{T}{\delta\sqrt{p}})$ queries to U. The detailed proof can be found in [9].*

B.4 Median Evaluation

Theorem 4 *Consider a unitary $U : |0^{\otimes m}\rangle \mapsto \sqrt{\alpha}|v, 1\rangle + \sqrt{1-\alpha}|g, 0\rangle$ for some $1/2 \leq \alpha \leq 1$ in time T. There exits a quantum algorithm that, for any $\Delta > 0$ and for any $1/2 < \alpha_0 \leq \alpha$, produce a state $|\psi\rangle$ such that $\||\psi\rangle - \left|0^{\otimes mL}\right\rangle|x\rangle\| \leq \sqrt{2\Delta}$ for some integer L in time*

$$2T\left\lceil\frac{\log(1/\Delta)}{2\left(|\alpha_0|-1/2\right)^2}\right\rceil. \tag{25}$$

Refer to [65] for a detailed proof.

C Additional clarification of the proposed model

C.1 Definition of the infinite-width GNN and multi-head attention

GNTK is a generalization of NTK from infinite-width fully connected neural network to graph neural network, which is a well-established approach in recent literature and thus we miss some detailed explanation in our paper for space saving (and also due to the complexity of the details which

otherwise will cost a lot of space), and the details can be found in [17]. Here we give an intuitive illustration.

Consider a general GNN with the neighborhood feature aggregation function

$$\hat{\mathbf{h}}_u^l := \sum_{v \in \mathcal{N}(u) \cup \{u\}} \mathbf{h}_v^{(l-1)},$$

and the central node feature update function (R fully connected layers)

$$\mathbf{h}_u^l := \sqrt{\frac{c_\sigma}{d_R^l}} \sigma \left(\mathbf{W}_R^l \sqrt{\frac{c_\sigma}{d_{R-1}^l}} \sigma \left(\mathbf{W}_{R-1}^l \cdots \sqrt{\frac{c_\sigma}{d_1^l}} \cdot \sigma \left(\mathbf{W}_1^l \hat{\mathbf{h}}_u^l \right) \right) \right),$$

where $\mathbf{h}_u^l$ denotes the feature vector of node u in the l-th layer, $\mathcal{N}(u)$ denotes the neighborhood of node u, $\mathbf{W}_r^l \in \mathcal{R}^{d_r^l \times d_{r+1}^l}$ is the weight matrix ($r = 1, \ldots, R$), c_σ is a scaling factor. The infinite-width of GNN means that the output dimension d_r^l of $\mathbf{W}_r^l$ goes to infinity for $r = 1, \ldots, R$ and $l = 1, \ldots, L$.

Apart from the infinite-width GNN, the infinite-width transformer, which is exploited by us to enhance the GNTK, has also been investigated in [26]. Here we briefly explain what an infinite-width transformer looks like. Consider a single attention layer

$$\mathbf{Q}^h = \mathbf{H}\mathbf{W}_Q^h, \mathbf{K}^h = \mathbf{H}\mathbf{W}_K^h, \mathbf{V}^h = \mathbf{H}\mathbf{W}_V^h,$$

$$\mathbf{G}^h = \frac{\mathbf{Q}^h {\mathbf{K}^h}^\top}{\sqrt{s}}, \widehat{\mathbf{H}}^h = \zeta\left(\mathbf{G}^h\right)\mathbf{V}^h,$$

in Eq. 10, where $\mathbf{H} \in \mathcal{R}^{n \times d}$ is the node feature matrix and we ignore the superscript and the subscript for simplicity. $\mathbf{Q}^h$, $\mathbf{K}^h$, $\mathbf{V}^h \in \mathcal{R}^{d \times d'}$ are weight matrices corresponding to Query, Key and Value, respectively. s is a scaling factor. The square matrix $\mathbf{G}^h \in \mathcal{R}^{n \times n}$ can be viewed as a matrix whose element corresponds to the similarity of each pair of nodes. Then the operation $\widehat{\mathbf{H}}^h = \zeta\left(\mathbf{G}^h\right)\mathbf{V}^h$ transforms the node features of the last layer to the next layer depending on the node similarity. For a multi-head attention (transformer) layer, the equation is given as

$$transformer(\mathbf{H}) = concat(\widehat{\mathbf{H}}^1 || \widehat{\mathbf{H}}^2 || \cdots || \widehat{\mathbf{H}}^H)\mathbf{W}_O,$$

where $\mathbf{W}_O \in \mathcal{R}^{Hd' \times d''}$ is the weight matrix. The infinite-width transformer means that the output dimension d' of $\mathbf{Q}^h$, $\mathbf{K}^h$, $\mathbf{V}^h$, the output dimension d'' of $\mathbf{W}_O$, and the number of heads H go to infinity.

C.2 Our contributions beyond the quantum speedup

Although the current scale of quantum hardware can not support the application of large-scale quantum algorithms, it still can not stop the widespread attention of academia and industry to quantum computing, especially in quantum machine learning [46, 19]. We hope that our model can provide technical guidance for future research, and better exploit the immense benefits of quantum computing even in a classic simulation condition.

Apart from quantum speedup, there are additional contributions of our proposed model, which have been illustrated in Line 66 to Line 80 in the introduction section. And we hope that this novel graph learning model can bridge the gap between graph neural tangent kernel methods and attention mechanism. Here we briefly give an illustration.

Better performance compared with state-of-art graph models. The numerical results in Tab. 1 and Tab. 2 demonstrate that our model (AttentionGNTK) outperforms GNTK on 7 out of 10 datasets, reflecting that the introduced infinite-width multi-head attention is useful and can better capture distinct properties between different graphs. Furthermore, we compare our model with another quantum-inspired graph learner QS-CNN. Our model surpasses QS-CNN on 4 out of 6 datasets and performs similarly on the rest 2 datasets. It shows the superiority of our model in quantum graph learning.

Table 4: Running time comparison between different models.

	MUTAG	NCI1	IMDB-B	IMDB-M
GIN	22 sec	67 min	19 min	24 min
GNTK	9 sec	18 min	4 min	7 min
Ours	14 sec	21 min	4 min	9 min

More robust compared with GNNs when the number of layers becomes larger. As illustrated in Tab. 5, when the number of layers is larger than 5 (but less than 10), Our model is more resistant to the oversmooth problem than GIN.

Fewer layers for reaching the peak of classification accuracy compared with GNTK. As illustrated in Tab. 6, our model (attentionGNTK) reaches the peak of classification accuracy when the number of layers is small, while GNTK needs more layers to reach, indicating that our model is easier to capture the global structure information of the graph.

D Additional Experiments and Discussion

D.1 Running time comparison

We supplement the training time of our model on four selected datasets and compare it with two other models. To make a fair comparison, we set the layers of all the models to 2. All the experiments are performed on a workstation with a single machine with 1TB memory, one physical CPU with 28 cores Intel® Xeon® W-3175X CPU @ 3.10GHz, and a single GPU (Nvidia Quadro RTX 8000). The results are shown in Tab. 4. Although the speedup introduced by the quantum algorithm depends on the quantum devices, it shows that our proposed model still has a computational overhead reduction when training on classic computers. The running time is slightly higher than that of GNTK which is a lack of attention mechanism. It is noticed that our model. The runtime of our model is apparently faster than that of GIN.

It is worth mentioning that the quadratic quantum speedup will be realized when the quantum hardware becomes more feasible.

D.2 Model sensitivity to the number of layers

In the main body of the paper, we report the best classification accuracy of the model when the the number of layers L is selected from $\{2, 4, 6, 8\}$. We compare the graph classification accuracy bewteen GIN and our model at the same number of layers and the results are given in Tab. 5 and Tab. 6. The number in parentheses in the table indicates the number of layers.

From Tab. 5, it is shown that our model (AttentionGNTK) is more robust compared with GIN when the number of layers becomes larger. The main reason is that an additional feature aggregation, e.g., the transformer, can slow down the convergence rate, which is consistent with the observations in [29] that connectivity enhancement can help wide and deep GNNs to avoid a discrepancy between prediction and the ground truth.

While in Tab. 6, the results empirically demonstrate that our model (attentionGNTK) reaches the peak of classification accuracy when the number of layers is small, while GNTK needs more layers to reach, indicating that our model is easier to capture the global structure information of the graph. This could be interpreted from the theoretical perspective. The transformer (Eq. 12) captured the semantic information between each pair of (connected and disconnected) nodes with similar features. Consider $\mathbf{G} = \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{s}}$ in Eq. 10, where $\mathbf{Q}$ and $\mathbf{K}$ are linear transformation of the node feature matrix and we ignore the superscript and the subscript for simplicity. The $\mathbf{G}$ can be viewed as a matrix whose element corresponds to the similarity of each pair of nodes. Then the operation $\widehat{\mathbf{H}} = \zeta(\mathbf{G})\mathbf{V}$ transforms the node features of the last layer to the next layer depending on the node similarity. This enables the model to make better use of the graph structure to transmit information and perceive topology information over long distances.

Table 5: Classification accuracy between GIN [67] and ours with respect to different layers.

	GIN(4)	Ours(4)	GIN(6)	Ours(6)	GIN(8)	Ours(8)
MUTAG	87.6 ± 6.2	89.1 ± 7.8	88.5 ± 5.6	90.0 ± 8.5	86.2 ± 6.4	88.4 ± 7.4
PROTEINS	75.5 ± 3.0	75.0 ± 4.1	74.3 ± 3.0	76.1 ± 3.8	72.8 ± 3.5	74.2 ± 4.4
PTC	62.8 ± 5.0	64.9 ± 5.3	62.0 ± 6.2	66.2 ± 5.1	61.2 ± 7.1	63.4 ± 6.6
NCI1	82.3 ± 3.6	84.1 ± 1.2	80.1 ± 2.4	83.8 ± 1.2	77.2 ± 3.3	82.3 ± 2.2
IMDB-B	73.2 ± 4.1	75.7 ± 2.8	74.4 ± 6.0	76.9 ± 4.3	72.1 ± 5.2	75.1 ± 4.0
IMDB-M	51.7 ± 3.7	52.0 ± 4.1	52.0 ± 2.6	51.9 ± 3.7	48.2 ± 4.3	50.3 ± 4.5

Table 6: Classification accuracy between GNTK [17] and ours with respect to different layers.

	GNTK(4)	Ours(4)	GNTK(6)	Ours(6)	GNTK(8)	Ours(8)
PTC	62.9 ± 7.2	64.9 ± 5.3	63.5 ± 6.8	$\mathbf{66.2 \pm 5.1}$	65.2 ± 7.9	63.4 ± 6.6
NCI1	83.6 ± 2.1	$\mathbf{84.1 \pm 1.2}$	84.0 ± 0.9	83.8 ± 1.2	82.9 ± 1.8	82.3 ± 2.2