Supplementary Material

Jiongshu Wang[1], Jing Yang[3], Jiankang Deng[2], Hatice Gunes[3], Siyang Song[1,3*]

[1*]School of Computing and Mathematical Sciences, University of Exeter, Exeter, EX4 4PY, Devon, United Kingdom.

[2]Department of Computing, Imperial College London, London, SW7 2AZ, London, United Kingdom.

[3*]Department of Computer Science and Technology, University of Cambridge, Cambridge, CB3 0FD, Cambridgeshire, United Kingdom.

*Corresponding author(s). E-mail(s): s.song@exeter.ac.uk;
Contributing authors: jiongshuwang@gmail.com; y.jing2016@gmail.com;
jiankangdeng@gmail.com; Hatice.Gunes@cl.cam.ac.uk;

1 Differentiation Analysis of the GIG Layer

In this section, we provide the back-propagation analysis of the proposed GIG layer. Supposing that the loss value $\mathcal{L}$ is computed by the objective function $\updownarrow$ and $\overline{\mathcal{G}}(\overline{S}_G(\overline{V}_S, \overline{E}_S), \overline{P}_g, \overline{E}_P, \overline{E}_G)$ is the output of the GIG layer, the gradient of the proposed GIG layer w.r.t. the input GIG sample $\mathcal{G}(S_G(V_S, E_S), P_l, P_g, E_P, E_G)$ can be defined as:

$$\frac{\partial \mathcal{L}}{\partial \mathcal{G}(S_G(V_S, E_S), P_l, P_g, E_P, E_G)} = \frac{\partial \mathcal{L}}{\partial V_S} + \frac{\partial \mathcal{L}}{\partial E_S} + \frac{\partial \mathcal{L}}{\partial P_l} + \frac{\partial \mathcal{L}}{\partial P_g} + \frac{\partial \mathcal{L}}{\partial E_P} + \frac{\partial \mathcal{L}}{\partial E_G} \tag{1}$$

There are two types of directed proxy edges, either starting from graph vertices to local proxy vertex or starting from global proxy vertex to graph vertices. Here, we use E_P^L and E_P^G to denote these two kinds of proxy edges, respectively. Besides, P_l represents local proxy vertex and P_g represents global proxy vertex, respectively. Consequently, the $\frac{\partial \mathcal{L}}{\partial E_S}$, $\frac{\partial \mathcal{L}}{\partial V_S}$, $\frac{\partial \mathcal{L}}{\partial P_l}$, $\frac{\partial \mathcal{L}}{\partial P_g}$, $\frac{\partial \mathcal{L}}{\partial E_P^L}$, $\frac{\partial \mathcal{L}}{\partial E_P^G}$ and $\frac{\partial \mathcal{L}}{\partial E_G}$ can be computed as:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial E_S} &= \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \overline{E}_G} \frac{\partial \overline{E}_G}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \overline{E}_G} \frac{\partial \overline{E}_G}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \overline{E}_G} \frac{\partial \overline{E}_G}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \overline{E}_G} \frac{\partial \overline{E}_G}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{V}_S} \frac{\partial \hat{V}_S}{\partial \hat{E}_S} \frac{\partial \hat{E}_S}{\partial E_S}
\end{aligned} \tag{2}$$

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial E_P^L} &= \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \overline{E}_G} \frac{\partial \overline{E}_G}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial E_P^L} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial E_P^L} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \overline{E}_G} \frac{\partial \overline{E}_G}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial E_P^L} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial \overline{P}_g} \frac{\partial \overline{P}_g}{\partial \hat{P}_l} \frac{\partial \hat{P}_l}{\partial \hat{E}_P^S} \frac{\partial \hat{E}_P^S}{\partial E_P^L}
\end{aligned} \tag{3}$$

$$\frac{\partial \mathcal{L}}{\partial E_P^G} = \frac{\partial \mathcal{L}}{\partial \overline{V}_S} \frac{\partial \overline{V}_S}{\partial \overline{E}_P^G} \frac{\partial \overline{E}_P^G}{\partial E_P^G} \tag{4}$$

Table 1: Overview of the graph learning datasets involved in this work with their respective prediction levels, prediction tasks, and metrics.

Dataset	# Graphs	Avg. # nodes	Avg. # edges	Prediction level	Prediction task	Metric
ZINC-12K	12000	23.2	49.8	graph	regression	Mean Abs. Error
ZINC-full	250000	23.2	49.8	graph	regression	Mean Abs. Error
AQSOL	10000	17.6	35.8	graph	regression	Mean Abs. Error
MNIST	70000	70.6	564.5	graph	10-class classif.	Accuracy
CIFAR10	60000	117.6	941.1	graph	10-class classif.	Accuracy
ENZYMES	600	32.63	62.14	graph	6-class classif.	Accuracy
PROTEINS	1113	39.06	3.70	graph	2-class classif.	Accuracy
D&D	1,178	284.32	5.0	graph	2-class classif.	Accuracy
OGBG-PPA	158,100	243.4	2,266.1	graph	37-class classif.	Accuracy
PATTERN	14000	117.5	4749.2	inductive node	binary classif.	Accuracy
CLUSTER	12000	117.2	4301.7	graph	6-class classif.	Accuracy
TSP	12000	275.8	6,894.0	edge	6-class classif.	F1 Score

Table 2: Statistics of the two employed human skeleton video-based action recognition datasets.

Dataset	# clips	Avg. # Frames	Avg. # Joints (Nodes)	Prediction level	Prediction task	Metric
NTU X-Sub	40000	300	37.5	graph	10-class classif.	Accuracy
NTU X-View	38000	300	37.5	graph	10-class classif.	Accuracy

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial P_l} = &\frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial P_l} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial P_l} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial P_l} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial P_l} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial P_l} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial P_l} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial P_l} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial P_l} \qquad (5)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial P_g} = &\frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial P_g} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial P_g} \qquad (6)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial E_G} = &\frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial E_G} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial E_G} \qquad (7)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial V_S} = &\frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} + \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial \hat{E}_S}\frac{\partial \hat{E}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial \hat{E}_S}\frac{\partial \hat{E}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \overline{E}_G}\frac{\partial \overline{E}_G}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{E}_P^S}\frac{\partial \hat{E}_P^S}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \\
&+ \frac{\partial \mathcal{L}}{\partial \overline{V}_S}\frac{\partial \overline{V}_S}{\partial \overline{E}_P^G}\frac{\partial \overline{E}_P^G}{\partial \overline{P}_g}\frac{\partial \overline{P}_g}{\partial \hat{P}_l}\frac{\partial \hat{P}_l}{\partial \hat{V}_S}\frac{\partial \hat{V}_S}{\partial V_S} \qquad (8)
\end{aligned}
$$

Table 3: Runtime analysis of our best GIG-GatedGCN model on seven graph datasets.

Model	Type	MNIST	CIFAR10	ZINC	AQSOL	PATTERN	CLUSTER	TSP
GIG-GatedGCN	Each iteration	0.35s	0.34s	0.21s	0.26s	0.60s	0.47s	0.31s
	Each epoch	2.50m	1.97m	0.28m	0.26m	0.67m	1.23m	1.91m
	Convergence	3.58h	3.78h	6.69h	0.20h	0.48h	1.47h	1.08h
	Inference	0.0026s	0.0025s	0.0016s	0.0025s	0.0025s	0.0042s	0.0090s
GatedGCN	Each iteration	0.23s	0.20s	0.14s	0.11s	0.11s	0.23s	0.55s
	Each epoch	1.66m	1.19m	0.18m	0.11m	0.56m	1.12m	1.44m
	Convergence	3.55h	4.15h	0.62h	0.41h	3.96h	6.81h	2.85h
	Inference	0.0028s	0.0026s	0.0017s	0.0032s	0.0024s	0.0045s	0.0090s

Table 4: Training settings of our best models for 12 generic graph analysis datasets and 2 human-skeleton based action recognition datasets.

Dataset	#Layer	Learning rate	Loss	Decay strategy	Optimizer
MNIST	3	0.0018	Cross Entropy	CosineAnnealing	AdamW
CIFAR10	3	0.0018	Cross Entropy	CosineAnnealing	AdamW
PROTEINS	3	0.0012	Cross Entropy	CosineAnnealing	AdamW
D&D	3	0.0012	Cross Entropy	CosineAnnealing	AdamW
ENZYMES	3	0.0012	Cross Entropy	CosineAnnealing	AdamW
OGBG-PPA	1	0.01	Cross Entropy	-	Adam
ZINC	3	0.0025	L1	CosineAnnealing	AdamW
ZINC-full	3	0.0025	L1	CosineAnnealing	AdamW
AQSOL	1	0.0025	L1	CosineAnnealing	AdamW
CLUSTER	1	0.0027	Cross Entropy	ReduceLROnPlateau	AdamW
PATTERN	2	0.0030	Cross Entropy	ReduceLROnPlateau	AdamW
TSP	3	0.002	Cross Entropy	ReduceLROnPlateau	AdamW
NTU X-Sub	1	0.1	Cross Entropy	MultiStep	AdamW
NTU X-View	1	0.1	Cross Entropy	MultiStep	AdamW

2 Model complexity analysis for example GIG-GatedGCN and GIG-GAT networks and runtime Analysis

A GIG network consists of a GSG layer, a set of GIG hidden layers and a GIG output layer. As discussed in the main document, the overall time complexity of the **GSG layer** is $O(|\mathcal{G}| \times max(V_S) \times n + |\mathcal{G}|^2 \times n)$, where $|\mathcal{G}|$ represents the number of GIG vertices, $max(V_S)$ indicates the maximal number of graph vertices in all GIG vertices, and n denotes the constant time operation. After that, each **GIG hidden layer** is made up of a GVU and a GGU module, whose specific time complexity is determined by the employed edge/vertex updating algorithms.

- **For GIG-GatedGCN**, the total time complexity of the GVU and GGU are both $O(E \times D_e + V \times D_v)$, where E indicates the number of edges in the graph, D_e indicates the dimensionality of the features being processed for edges, V indicates the number of vertices in the graph and D_v indicates the dimensionality of the features being processed for vertices. Consequently, the time complexity of each GIG-GatedGCN hidden layer can be defined as $O(2 \times (E \times D_e + V \times D_v))$.
- **For GIG-GAT**, the total time complexity of the GVU and GGU can be defined as $O(H \times (E \times D_e + V \times D_v))$, where H indicates the number of attention heads, E indicates the number of edges in the graph, D_e indicates the dimensionality of the features being processed for edges, V indicates the number of vertices in the graph and D_v indicates the dimensionality of the features being processed for vertices. Then, the time complexity of each GIG-GAT hidden layer can be defined as $O(2 \times H \times (E \times D_e + V \times D_v))$.

The **GIG output layer** additionally updates graph vertices that are not connected with global proxy vertices as well as graph edges compared to GIG hidden layer. As a result, the total time complexity of each GIG output layer can be defined as $O(E \times D_e + V \times D_v)$ for GIG-GatedGCN and $O(H \times (E \times D_e + V \times D_v))$ for GIG-GAT. As a result, the entire model complexity for a GIG-GatedGCN network that contains k hidden layers can be defined as $O(|\mathcal{G}| \times max(V_S) \times n + |\mathcal{G}|^2 \times n + (2 \times k + 1) \times (E \times D_e + V \times D_v))$. Meanwhile, the model complexity of a GIG-GatedGCN network that contains k hidden layers can be defined as $O(|\mathcal{G}| \times max(V_S) \times n + |\mathcal{G}|^2 \times n + (2 \times k + 1) \times H \times (E \times D_e + V \times D_v))$.

Table 3 and Table 6 further provide the runtime analysis for our GIG networks on various datasets in

Table 5: Statistical significant difference analysis (T-Test) results between our best GIG systems and the corresponding second best systems for both tasks as well as several ablation studies, where the confidence interval is set to 0.05 (+ denotes there is significant difference while - denotes there is no significant difference).

Type	Significant difference (P-value)
GIG vs EGT	+ (0.0540)
GIG vs InfoGCN	- (0.1034)
GIG vs GatedGCN	+ (0.3238)
GIG vs CTR-GCN	+ (0.1682)
(i) GSG+GVU+GGU vs GSG+GVU	+ (0.0280)
(ii) Default number of proxy edge vs Connect all graph vertices	+ (0.0018)
(iii) Proxy edge default setting vs Most similar connections	+ (0.0020)
(iv) GIG edge default setting vs Most similar connections	+ (0.0003)
(v) Averaging graph vertices vs Random graph vertex	+ (0.0047)
(vi) GVU → GGU vs GGU → GVU	+ (0.2142)

terms of training and inference, respectively. While our GIG-GatedGCN has more time consumption of each training iteration/epoch than original GatedGCN, it has much less convergence time consumption on five out of seven datasets and very similar inference speed as the original GatedGCN. In addition, the superior performance of GIG-CTR-GCN is at the cost of longer training time. However, its inference speed is not slower than the original CTR-GCN.

Influences of the numbers of proxy edges and GIG edges on the GVU and GGU's complexity: In GVU, the number of proxy edge can be defined as $|E_P^i|/2$. We use $O(f_E^{\text{GVU}})$ to represent the algorithmic complexity of edge processing involved in the specific algorithm used by the GVU module. Then the time complexity related to the number of proxy edges can be defined as $O(|E_P^i|/2 \times f_E^{\text{GVU}})$. In GGU, the number of proxy edge is same as before, which can be defined as $|E_P^i|/2$. Continuously, the number of GIG edges can be defined as $|\mathcal{N}_{\text{global}}(P)| \times |P|$. Furthermore, we can use $O(f_E^{\text{GGU}})$ to represent the algorithmic complexity of edge processing utilised by GGU module. Therefore, there will be time complexity in GGU module as $O(f_E^{\text{GGU}} \times (|E_P^i|/2 + |\mathcal{N}_{\text{global}}(P)| \times |P|))$ in total. Overall, the main time complexity occupation in both GVU and GGU can be calculated as $O(|E_p^i|/2 \times f_E^{\text{GVU}} + (|E_P^i|/2 + |\mathcal{N}_{\text{global}}(P)| \times |P|) \times f_E^{\text{GGU}})$. From information extracted from above, the computational complexity of model during training and inference is closely related to the number of proxy edges and GIG edges. From a large scale of experiments, the strategy we choose to construct connections between local/-global proxy vertices and graph vertices is to partially connect each local/global proxy vertex with graph vertices (e.g. 10% graph vertices of each GIG vertex in this paper) in the corresponding GIG vertex and the strategy utilized to build connections between different global proxy vertices with local proxy vertices from other GIG vertices is upon the KNN algorithm, which is generally no more than 9 neighbours in all downstream tasks. Then the actual extra model complexity related to proxy edge and GIG edge during running can be limited in a small range. In particular, the time complexity discussed above is only related to the FLOPs but all parameters are shared in GVU and GGU modules (e.g., proxy edge and graph edge will utilise same parameters) for different types of edges.

Table 6: Runtime analysis of our best GIG-CTR-GCN model on two skeleton video-based action recognition datasets.

Model	Type	NTU X-Sub	NTU X-View
GIG-CTR-GCN	Each iteration	1.20s	1.80s
	Each epoch	11.37m	16.00m
	Convergence	8.72h	15.73h
	Inference	0.0131s	0.0152s
CTR-GCN	Each iteration	0.69s	1.17s
	Each epoch	7.22m	11.50m
	Convergence	7.58h	12.46h
	Inference	0.0135s	0.0154s

3 Details of the two action recognition datasets

Table 2 further provides the details of the employed human skeleton video-based action recognition datasets.

4 Training hyper-parameter settings for all experiments

We provide detailed training settings for generic graph analysis tasks and skeleton-based video action recognition tasks in Table 4 which represent the settings for our best models. Please check the code for more details. All models are optimised using the AdamW [1] optimizer.

5 Effects of different proportions of similar and non-similar GIG vertices to be connected with each global proxy vertex

Table 7: Ablation study for different proportions in construction of GIG edges. The (i) represents the proportion of similar GIG vertices adopted and the (ii) represents the proportion of non-similar GIG vertices adopted.

Strategy		MNIST	PATTERN	TSP
(i)	(ii)	Test Acc(%) ↑		Test F_1 ↑
10%	90%	98.76 ± 0.04	86.80 ± 0.05	0.857 ± 0.001
20%	80%	98.75 ± 0.03	86.81 ± 0.03	0.860 ± 0.001
30%	70%	98.77 ± 0.03	86.83 ± 0.02	0.859 ± 0.002
40%	60%	98.75 ± 0.04	86.85 ± 0.01	0.859 ± 0.001
50%	50%	$\mathbf{98.80 \pm 0.03}$	$\mathbf{86.87 \pm 0.01}$	$\mathbf{0.863 \pm 0.001}$
60%	40%	98.79 ± 0.01	86.83 ± 0.03	0.862 ± 0.001
70%	30%	98.73 ± 0.05	86.82 ± 0.01	0.862 ± 0.003
80%	20%	98.79 ± 0.03	86.85 ± 0.02	0.859 ± 0.001
90%	10%	98.71 ± 0.07	86.79 ± 0.03	0.857 ± 0.006

Table 7 presents the ablation studies as comparing the different combination of proportions of similar and non-similar GIG edges. The results show that 50% similar GIG edges and 50 % non-similar global-edges possesses the best performance.

6 Statistical difference analysis

We provide statistical significant difference analysis results in Table 5, where we first compare our best GIG-based system with the second best systems for seven generic graph analysis tasks and two human skeleton-based video action recognition tasks, respectively. It can be seen that our GIG significantly better than the second best system (i.e., EGT) in generic graph analysis while also achieving promising p-value results compared to the InfoGCN on skeleton-based action recognition. Furthermore, the p-value results for GIG and its baselines, including GatedGCN and CTR-GCN, show the significant difference as well.

Meanwhile, it also compares our best systems with the second best variants for six ablation study experiments, which are: (i) the impacts of individual modules in GIG; (ii) different number of proxy edges; (iii) proxy edge definition strategies; (iv) GIG edge definition strategies; (v) local proxy vertex initialization strategies; and (vi) the impacts of the order of different modules.

7 Analysis of different similarity calculation strategies

Table 8 compares the results achieved by using three different metrics to measure the similarity between (i) the local proxy vertex and its corresponding graph vertices for defining proxy edges; and (ii) local proxy vertices and global proxy vertices for defining GIG edges. It is clear that the employed cosine similarity achieved marginal but consistent advantages over others, i.e., our approach is relatively robust to the employed similarity measurement.

Table 8: Results achieved for different similarity calculation strategies used for measuring the similarity between: (i) the local proxy vertex and its corresponding graph vertices for defining proxy edges; and (ii) local proxy vertices and global proxy vertices for defining GIG edges.

Strategy	MNIST	PATTERN	TSP
-	Test Acc(%) ↑		Test F_1 ↑
L1 norm	98.64 ± 0.03	86.57 ± 0.09	0.854 ± 0.003
L2 norm	98.76 ± 0.05	86.81 ± 0.06	0.861 ± 0.002
Cosine similarity	$\mathbf{98.80 \pm 0.03}$	$\mathbf{86.87 \pm 0.01}$	$\mathbf{0.863 \pm 0.001}$

8 Baseline and Comparative Models Evaluated on the PeMSD7 Dataset for Traffic Forecasting

All approaches described below have been evaluated on the PeMSD7 dataset to verify their performances [2].

- **(i)** Historical Average (HA) [3] is a simple baseline that predicts future traffic conditions by averaging historical values at the same time periods (e.g., same time of day, same day of week). This is a common baseline that have been widely used in traffic forecasting benchmarks for performance comparisons [2, 4, 5];
- **(ii)** Linear Support Vector Regression (LSVR) [6] applies Support Vector Machine (SVM) to regression tasks by finding an optimal hyperplane that fits the data within an acceptable error margin. It have been frequently applied to spatial-temporal traffic data modeling [4];
- **(iii)** Auto-Regressive Integrated Moving Average (ARIMA) [7] is a classical statistical time series forecasting model that combines auto-regression, differencing, and moving average components. It is a standard statistical solution specifically designed for time series forecasting [8–10] and have been evaluated in the PeMSD7 dataset [2];
- **(iv)** Full-Connected LSTM (FC-LSTM) [11] employs Long Short-Term Memory networks with fully connected layers to capture temporal dependencies in sequential data. This deep learning approach is specifically designed for time series forecasting and can model complex temporal patterns, making it an important baseline for temporal modeling. This approach has been frequently utilized in traffic forecasting tasks [12];

Table 9: The numbers and proportions of GIG edges under our best-performing models.

Task	Graph classification						
Dataset	MNIST	CIFAR10	ENZYMES	DD	PROTEINS	OGBG-PPA	NTU RGB+D
GIG edge number (starting from each GIG sample's local proxy vertex)	6	6	6	6	6	6	9
GIG edge proportion (for each GIG sample's proxy vertex)	4.68%	4.68%	4.68%	4.68%	4.68%	4.68%	3.52%
GIG vertex number	128	128	128	128	128	128	256

Task	Graph regression					Node classification		Edge classification
Dataset	ZINC(10K)	ZINC-full	AQSOL	PeMSD7(M)	PeMSD7(L)	PATTERN	CLUSTER	TSP
GIG edge number (each GIG vertex)	6	9	6	4	4	6	9	3
GIG edge proportion (each GIG vertex)	18.75%	7.03%	18.75%	3.16%	3.16%	9.38%	14.06%	9.38%
GIG vertex number	32	128	32	128	128	64	64	32

Table 10: Ablation study results achieved for different GIG edge proportions.

Datasets	MNIST	PATTERN	TSP
Proportion	Test Acc(%) ↑		Test F_1 ↑
5%	$\mathbf{98.80 \pm 0.03}$	86.85 ± 0.01	0.862 ± 0.001
10%	98.78 ± 0.04	$\mathbf{86.87 \pm 0.01}$	$\mathbf{0.863 \pm 0.002}$
30%	98.75 ± 0.04	86.83 ± 0.04	$\mathbf{0.863 \pm 0.001}$
50%	98.75 ± 0.03	86.81 ± 0.08	$\mathbf{0.863 \pm 0.002}$
70%	98.74 ± 0.04	86.78 ± 0.05	0.862 ± 0.006
100%	98.72 ± 0.05	86.77 ± 0.06	0.862 ± 0.008

- **(v)** Graph Convolutional GRU (GCGRU) [5] combines Graph Convolutional Networks with Gated Recurrent Units, designed to handle both the spatial relations (road network structure) and temporal dynamics of traffic data simultaneously. This represents a more advanced spatial-temporal deep learning approach, and have been adopted widely to process graph-structured spatial-temporal time-series data, including other traffic monitoring dataset, such as PeMSD4 and PeMSD8 [13, 14];
- **(vi)** Spatial-Temporal Graph Convolutional Network (STGCN) [2] is specifically designed for spatial-temporal graph data, using specialized convolution operations to capture both spatial dependencies between nodes in the traffic network and temporal dynamics. This method has been widely utilized in the traffic forecasting tasks, including the Xiamen, PeMSD4, PeMSD7 and PeMSD8, etc [4, 5, 14]. It's included as a direct predecessor to the proposed GIG-STGCN model, providing a clear reference point for measuring the improvements gained from our GIG approach;
- **(vii)** SkateFormer [15] represents a transformer-based approach for spatial-temporal graph modelling, incorporating attention mechanisms to model dependencies between different vertices and time steps. As one of the most recent state-of-the-art solution, it provides a contemporary benchmark against our the GIG-STGCN model.

9 The number and proportion of GIG edges

Table 9 summarises the number and relative proportions of GIG edges belonging to each GIG vertex for each dataset. Since the connections between GIG vertices are constructed by linking the local proxy vertex of each GIG vertex to k global proxy vertices belonging to k other GIG vertices, we define the GIG edge proportion as:

$$\rho = \frac{k}{|S_G| - 1}. \tag{9}$$

where $|S_G|$ indicates the total number of GIG vertices of a GIG sample. As summarised in Table 9, we defined 6 GIG edges per GIG vertex ($\rho \approx 5\%$) for graph classification datasets, whereas 9 GIG edges ($\rho \approx 4\%$) are defined for each GIG vertex in NTU RGB+D dataset. For graph regression datasets, the number of GIG edges per GIG vertex ranges from 4 to 9 ($\rho \approx 3\%$–19%). For node classification datasets, we define 6-9 GIG edges ($\rho \approx 9\%$-14%) for each GIG vertex, while 3 GIG edges ($\rho \approx 10\%$) are defined for each GIG vertex in the edge classification TSP dataset.

10 Impacts of different GIG edge proportions

Table 10 reports the results achieved for different proportions, demonstrating that our GIG approach is robust to variations in GIG edge proportions/numbers. **For the MNIST dataset**, the accuracy consistently remains above 98.7%, with the best performance achieved at a 5% edge proportion. **For the PATTERN dataset**, accuracy is highly stable across all proportions, with the performance peak at 10% proportion. **For the TSP dataset**, the model maintains high F1 scores throughout, with optimal results observed at 10%, 30%, and 50% proportions, followed by only slight declines at larger proportions. In summary, the findings above indicate that GIG performance is largely invariant to the number of GIG edges, which consistently achieved decent results across diverse edge settings.

References

[1] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)

[2] Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 3634–3640 (2018)

[3] Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., Li, H.: T-gcn: A temporal graph convolutional network for traffic prediction. IEEE transactions on intelligent transportation systems **21**(9), 3848–3858 (2019)

[4] Zheng, C., Fan, X., Wang, C., Qi, J.: Gman: A graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1234–1241 (2020)

[5] Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: International Conference on Learning Representations (2018)

[6] Drucker, H., Burges, C.J., Kaufman, L., Smola, A., Vapnik, V.: Support vector regression machines. Advances in neural information processing systems **9** (1996)

[7] Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time Series Analysis: Forecasting and Control. John Wiley & Sons, ??? (2015)

[8] Williams, B.M., Hoel, L.A.: Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. Journal of transportation engineering **129**(6), 664–672 (2003)

[9] Smith, B.L., Williams, B.M., Oswald, R.K.: Comparison of parametric and nonparametric models for traffic flow forecasting. Transportation Research Part C: Emerging Technologies **10**(4), 303–321 (2002)

[10] Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C.: Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach. Transportation Research Part C: Emerging Technologies **13**(3), 211–234 (2005)

[11] Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. Advances in neural information processing systems **27** (2014)

[12] Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C: Emerging Technologies **54**, 187–197 (2015)

[13] Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph wavenet for deep spatial-temporal graph modeling. arXiv preprint arXiv:1906.00121 (2019)

[14] Bai, L., Yao, L., Li, C., Wang, X., Wang, C.: Adaptive graph convolutional recurrent network for traffic forecasting. Advances in neural information processing systems **33**, 17804–17815 (2020)

[15] Do, J., Kim, M.: Skateformer: skeletal-temporal transformer for human action recognition. In: European Conference on Computer Vision, pp. 401–420 (2024). Springer