Paper

Do Transformers Really Perform Bad for Graph Representation?

The Transformer architecture has become a dominant choice in many domains, such as natural language processing and computer vision. Yet, it has not achieved competitive performance on popular leaderboards of graph-level prediction compared to mainstream GNN variants. Therefore, it remains a mystery how Transformers could perform well for graph representation learning. In this paper, we solve this mystery by presenting Graphormer, which is built upon the standard Transformer architecture, and could attain excellent results on a broad range of graph representation learning tasks, especially on the recent OGB Large-Scale Challenge. Our key insight to utilizing Transformer in the graph is the necessity of effectively encoding the structural information of a graph into the model. To this end, we propose several simple yet effective structural encoding methods to help Graphormer better model graph-structured data. Besides, we mathematically characterize the expressive power of Graphormer and exhibit that with our ways of encoding the structural information of graphs, many popular GNN variants could be covered as the special cases of Graphormer.

Table 1: Results on PCQM4M-LSC. * indicates the results are cited from the official leaderboard Hu et al. (2021).

method	#param.	train MAE	validate MAE
GCN Kipf and Welling (2016)	2.0M	0.1318	0.1691 (0.1684*)
GIN Xu et al. (2019)	3.8M	0.1203	0.1537 (0.1536*)
GCN-VN Kipf and Welling (2016); Gilmer et al. (2017)	4.9M	0.1225	0.1485 (0.1510*)
GIN-VN Xu et al. (2019); Gilmer et al. (2017)	6.7M	0.1150	0.1395 (0.1396*)
GINE-VN Brossard et al. (2020); Gilmer et al. (2017)	13.2M	0.1248	0.1430
DeeperGCN-VN Li et al. (2020a); Gilmer et al. (2017)	25.5M	0.1059	0.1398
GT Dwivedi and Bresson (2021)	0.6M	0.0944	0.1400

Table 3: Results on MolHIV.

Parse references

method	#param.	AP (%)
DeeperGCN-VN+FLAG Li et al. (2020a)	5.6M	28.42±0.43
DGN Beaini et al. (2021)	6.7M	28.85±0.30
GINE-VN Brossard et al. (2020)	6.1M	29.17±0.15
PHC-GNN Le et al. (2021)	1.7M	29.47±0.26
GINE-APPNP Brossard et al. (2020)	6.1M	29.79±0.30
GIN-VNXu et al. (2019) (fine-tune)	3.4M	29.02±0.17
Graphormer-FLAG	119.5M	**31.39±**0.32

Table 3: Results on MolHIV.

Parse references

method	#param.	AUC (%)
GCN-GraphNorm Brossard et al. (2020); Cai et al. (2021)	526K	78.83±1.00
PNA Corso et al. (2020)	326K	79.05±1.32
PHC-GNN Le et al. (2021)	111K	79.34±1.16
DeeperGCN-FLAG Li et al. (2020a)	532K	79.42±1.20
DGN Beaini et al. (2021)	114K	79.70±0.97
GIN-VNXu et al. (2019) (fine-tune)	3.3M	77.80±1.82
Graphormer-FLAG	47.0M	**80.51±**0.53

Table 4: Results on ZINC.

Parse references

method	#param.	test MAE
GIN Xu et al. (2019)	509,549	0.526±0.051
GraphSage Hamilton et al. (2017)	505,341	0.398±0.002
GAT Veličković et al. (2018)	531,345	0.384±0.007
GCN Kipf and Welling (2016)	505,079	0.367±0.011
GatedGCN-PE Bresson and Laurent (2017)	505,011	0.214±0.006
MPNN (sum) Gilmer et al. (2017)	480,805	0.145±0.007
PNA Corso et al. (2020)	387,155	0.142±0.010

Table 5: Ablation study results on PCQM4M-LSC dataset with different designs.

Parse references

Node Relation Encoding		Centrality	Edge Encoding			valid MAE
Laplacian PEDwivedi and Bresson (2021)	Spatial	Centrality	via node	via Aggr	via attn bias(Eq.7)	valid MAE
-	-	-	-	-	-	0.2276
✓	-	-	-	-	-	0.1483
-	✓	-	-	-	-	0.1427
-	✓	✓	-	-	-	0.1396
-	✓	✓	✓	-	-	0.1328
-	✓	✓	-	✓	-	0.1327
-	✓	✓	-	-	✓	0.1304

Table 6: Statistics of the datasets.

Dataset	Scale	# Graphs	# Nodes	# Edges	Task Type
PCQM4M-LSC	Large	3,803,453	53,814,542	55,399,880	Regression
OGBG-MolPCBA	Medium	437,929	11,386,154	12,305,805	Binary classification
OGBG-MolHIV	Small	41,127	1,048,738	1,130,993	Binary classification
ZINC (sub-set)	Small	12,000	277,920	597,960	Regression

Table 7: Model Configurations and Hyper-parameters of Graphormer on PCQM4M-LSC.

	Graphormer\textsc*Small*	Graphormer
#Layers	6	12
Hidden Dimension *d*	512	768
FFN Inner-layer Dimension	512	768
#Attention Heads	32	32
Hidden Dimension of Each Head	16	24
FFN Dropout	0.1	0.1
Attention Dropout	0.1	0.1

Peak Learning Rate	3e-4	2e-4
Batch Size	1024	1024
Warm-up Steps	60K	60K
Learning Rate Decay	Linear	Linear
Adam ϵ	1e-8	1e-8
Adam (β1, β2)	(0.9, 0.999)	(0.9, 0.999)
Gradient Clip Norm	5.0	5.0
Weight Decay	0.0	0.0

Table 8: Hyper-parameters for Graphormer on OGBG-MolPCBA, where the text in bold denotes the hyper-parameters we eventually use.

	Graphormer
Max Epochs	{2, 5, **10**}
Peak Learning Rate	{2e-4, **3e-4**}
Batch Size	256
Warm-up Ratio	0.06
Attention Dropout	0.3
m	{1, 2,3,**4**}
a	0.001
ϵ	0.001

Table 9: Hyper-parameters for Graphormer on OGBG-MolHIV, where the text in bold denotes the hyper-parameters we eventually use.

	Graphormer
Max Epochs	8
Peak Learning Rate	2e-4
Batch Size	128
Warm-up Ratio	0.06
Dropout	0.1
Attention Dropout	0.1

Table 10: Model Configurations and Hyper-parameters on ZINC(sub-set).

	Graphormer\textsc*Slim*
#Layers	12
Hidden Dimension	80
FFN Inner-Layer Hidden Dimension	80
#Attention Heads	8
Hidden Dimension of Each Head	10
FFN Dropout	0.1
Attention Dropout	0.1
Embedding Dropout	0.0
Max Steps	400*K*
Max Epochs	10*K*
Peak Learning Rate	2e-4
Batch Size	256
Warm-up Steps	40*K*
Learning Rate Decay	Linear
Adam ϵ	1e-8
Adam (β1, β2)	(0.9, 0.999)
Gradient Clip Norm	5.0
Weight Decay	0.01

Table 11: Hyper-parameters for fine-tuning GROVER on MolHIV and MolPCBA.

	GROVER	GROVER\textsc*LARGE*
Dropout	{0.1, 0.5}	{0.1, 0.5}
Max Epochs	{10, 30, 50}	{10, 30}
Learning Rate	{5e-5, 1e-4, 5e-4, 1e-3}	{5e-5, 1e-4, 5e-4, 1e-3}
Batch Size	{64, 128}	{64, 128}
Initial Learning Rate	1e-7	1e-7
End Learning Rate	1e-9	1e-9

method	#param.	AUC (%)
Morgan Finger Prints + Random Forest*	230K	**80.60±**0.10
GROVER*[46]	48.8M	79.33±0.09
GROVER\textsc*Large**[46]	107.7M	80.32±0.14
Graphormer-FLAG	47.0M	80.51±0.53

Table 13: Comparison to pre-trained Transformer-based GNN on MolPCBA. * indicates that additional features for molecule are used.

Parse references

method	#param.	AP (%)
GROVER*[46]	48.8M	16.77±0.36
GROVER\textsc*Large**[46]	107.7M	13.05±0.18
Graphormer-FLAG	47.0M	**31.39±**0.32

Results in Papers With Code