ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10447806

MERG: MULTI-DIMENSIONAL EDGE REPRESENTATION GENERATION LAYER FOR GRAPH NEURAL NETWORKS

Yuxin Song[1], *Cheng Luo*[2], *Aaron Jackson*[3], *Xi Jia*[4], *Weicheng Xie*[2], *Linlin Shen*[2], *Hatice Gunes*[5], *Siyang Song*[6,5∗]

[1] Baidu, China, [2] Shenzhen University, China, [3] University of Dundee, UK
[4] University of Birmingham, UK, [5] University of Cambridge, UK, [6] University of Leicester, UK
∗ Corresponding author: Siyang Song, ss2796@cam.ac.uk

ABSTRACT

Edges are essential in describing relationships among nodes. While existing graphs frequently use a single-value edge to describe association between each pair of node vectors, crucial relationships may be disregarded if they are not linearly correlated, which may limit graph analysis performance. Although some recent Graph Neural Networks (GNNs) can process graphs containing multi-dimensional edge features, they cannot convert single-value edge graphs to multi-dimensional edge graphs during propagation. This paper proposes a generic Multi-dimensional Edge Representation Generation (MERG) layer that can be inserted into any GNNs for heterogeneous graph analysis. It assigns multi-dimensional edge features for the input single-value edge graph, describing multiple task-specific and global context-aware relationship cues between each connected node pair. Results on eight graph benchmark datasets demonstrate that inserting the MERG layer into widely-used GNNs (e.g., GatedGCN and GAT) leads to major performance improvements, resulting in state-of-the-art (SOTA) results on seven out of eight evaluated datasets. Our code is publicly available at [1].

Index Terms— Multi-dimensional edge feature, Graph Neural Networks, Global contextual information

1. INTRODUCTION

Graphs have been widely used to represent various types of real-world data, such as human face [1], skeleton [2], 3D point cloud recognition [3], semantic segmentation [4], action recognition [5], research collaboration relationships [6], etc. A typical graph is made up of a set of nodes and edges, where each edge describes the relationship between a pair of connected nodes [7], deciding the message passing mechanism between them. However, most existing graph datasets or approaches only define edges' presences based on manually-defined rules (e.g., the euclidean distance between node [8] and node categories [9]), where the majority of them use single value to define the edge presence [2] or strength of association between a pair of nodes [10, 11]. Since manually defined single-value edges can only reflect a specific relationship between nodes, multiple crucial and task relationship cues would be ignored if each node feature pair are not linearly correlated, i.e., the message passing mechanism defined by single-value edges could not accurately exchange all task-related messages among nodes (**Problem 1**). Specifically, multiple features (i.e., multi-dimensional edge feature) are required to accurately describe the relationship between each pair of nodes and allow multiple task-specific messages to be exchanged among nodes during the propagation. This assumption (i.e., the effectiveness of the multi-dimensional edge feature) has been frequently validated in recent studies on various graph-related tasks [1, 8, 12, 13, 14, 15].

Although recent advanced GNNs [16, 17, 18, 19] can process graphs that contain multi-dimensional edge features, they are not able to directly encode a single-value edge graph to a multi-dimensional edge graph (i.e., *they can not assign multi-dimensional edge features to re-define edges of graphs whose edges are initially represented by a set of single value features*). This would limit existing GNNs' performances in analyzing graphs whose edge features are represented by single values. In summary, while the multi-dimensional edge features could more comprehensively and accurately describe the relationship between each pair of nodes in the graph and provide superior message passing mechanism during propagation, there is a lack of a effective and generic GNN layer that can not only assign multi-dimensional edge features to single-value edge graphs but also be directly inserted into various GNNs to allow them being trained in an end-to-end manner (**Problem 2**). Our experiments show that layer with such properties would consistently enhance GNN performance for various graph analysis tasks.

In this paper, we propose a generic Multi-dimensional Edge Representation Generation (MERG) layer that can be directly inserted into various GNNs, i.e., the MERG layers could be jointly trained with any GNN in an end-to-end manner. As a result, its weights would be optimized based on the target, and thus a well-trained MERG layer would learn

[1] https://github.com/SSYSteve/Learning-Graph-Representation-with-Task-specific-Topology-and-Multi-dimensional-Edge-Features

task-specific relationship cues from the input graph to encode its edge features. More specifically, our MERG layer can assign a multi-dimensional feature to each presented edge for any input graph regardless of its initial edge feature dimension (e.g., single-value edge graph), which represents multiple task-specific relationship cues between the corresponding node pair. Importantly, it takes both the relationship cues directly reflected by the corresponding node pair as well as their global contextual information into consideration. This enforces the learned multi-dimensional edge features to not only comprehensively represent local relationships between connected nodes but also have a global context-aware message passing mechanism.

2. THE PROPOSED APPROACH

Our MERG layer is the first plugin GNN layer that can convert single-value edge graphs to multi-dimensional edge graphs, where 'plugin' denotes that our MERG can be directly inserted into any GNNs. It consists of three main blocks: a local relationship modelling block (Sec. 2.1), a global contextual relationship modelling (GCRM) block (Sec. 2.2) and a local-global relationship fusion block (Sec. 2.3).

2.1. Local relationship modelling

Let the input graph have N vertices which are represented by K-dimensional vectors. To learn a task-specific multi-dimensional feature for a presented edge $\mathbf{\hat{e}_{i,j}}$, MERG first extracts local relationship cues (denoted as $\mathbf{e_{i,j}}^{\text{Local}}$) contained in corresponding nodes $\mathbf{v_i}$ and $\mathbf{v_j}$. As illustrated in Figure. 1, we use two learnable matrices (fully connected (FC) layers) W_1, $W_2 \in \mathbb{R}^{K \times D}$ to individually project $\mathbf{v_i}$ and $\mathbf{v_j}$ into a D-dimensional space, which are then concatenated as a single representation $\mathbf{e}^{\text{Lat}}_{\mathbf{i,j}} \in \mathbb{R}^{1 \times 2D}$ that contains both nodes' information. After that, a learnable matrix $W_l \in \mathbb{R}^{2D \times D}$ is employed to extract task-specific local relationship representation $\mathbf{e}^{\text{Local}}_{\mathbf{i,j}} \in \mathbb{R}^{1 \times D}$ from $\mathbf{e}^{\text{Lat}}_{\mathbf{i,j}}$. This can be formulated as:

$$\begin{aligned} \mathbf{e}^{\text{Lat}}_{\mathbf{i,j}} &= [\text{FC}(\mathbf{v}_i|W_1), \text{FC}(\mathbf{v}_j|W_2)] \\ \mathbf{e}^{\text{Local}}_{\mathbf{i,j}} &= \text{FC}(\mathbf{e}^{\text{Lat}}_{\mathbf{i,j}}|W_l) \end{aligned} \tag{1}$$

where the weight matrices W_1, W_2 and W_l are shared for all edge features' learning, and FC($|W$) denotes the FC layer whose projection is decided by the weight matrix W.

2.2. Global contextual relationship modelling

To allow edges forming a global context-aware message passing mechanism for the graph analysis, the MERG also extracts the global relationship cues $\mathbf{\hat{e}}^{\text{Ctx}}_{\mathbf{i,j}}$ between $\mathbf{v_i}$ and $\mathbf{v_j}$, which are reflected by the global context defined by all node features, i.e., the influences of $\mathbf{v_i}$ and $\mathbf{v_j}$ on the whole graph.

Let $\mathcal{V} \in \mathbb{R}^{N \times K}$ denote the matrix that concatenates all node features of the graph $\mathcal{G}$ based on their orders defined by the adjacency matrix $\mathcal{A}$, where N is the number of nodes (N rows), and K is the initial dimension of every node feature. To achieve $\mathbf{\hat{e}}^{\text{Ctx}}_{\mathbf{i,j}}$, the MERG first learns a latent global contextual representation $\mathbf{e}^{\text{Ctx}} \in \mathbb{R}^{N \times N \times D}$ from $\mathcal{V}$, which is formulated as:

$$\begin{aligned} \mathbf{e}^{\text{Ctx}} &= \bar{\mathbf{G_1}} \cdot \mathbf{G_2^T} \\ \textbf{Subject to } \ \mathbf{G_1} &= \text{FC}(\mathcal{V}|W_3) \ \ \mathbf{G_2} = \text{FC}(\mathcal{V}|W_4) \end{aligned} \tag{2}$$

where $\cdot$ denotes the matrix multiplication; $W_3 \in \mathbb{R}^{K \times D^2}$ and $W_4 \in \mathbb{R}^{K \times D}$ are also learnable matrices for producing latent global representations $\mathbf{G_1} \in \mathbb{R}^{N \times D^2}$ and $\mathbf{G_2} \in \mathbb{R}^{N \times D}$; and $\bar{\mathbf{G_1}} \in \mathbb{R}^{ND \times D}$ is reshaped from $\mathbf{G_1}$. Specifically, the W_4 projects each row vector of $\mathcal{V}$ from K dimensions to D dimensions, encoding a task-specific representation for each node, which is denoted as $\mathbf{G_2} \in \mathbb{R}^{N \times D}$. The W_3 projects $\mathcal{V}$ to a high-dimensional matrix $\mathbf{G_1} \in \mathbb{R}^{N \times D^2}$. Here, the i_{th} row of $\mathbf{G_1}$ has D^2 dimensions and is obtained by the weighted combination of all node features, which aims to encode i_{th} node's associations with others. Thus, both latent node features contained in $\bar{\mathbf{G_1}}$ and node relational latent features contained in $\mathbf{G_2}$ are learned in the context of global representation $\mathcal{V}$. We then conduct matrix multiplication between $\bar{\mathbf{G_1}}$ and $\mathbf{G_2^T}$ to obtain a global relationship representation $\mathbf{e}^{\text{Ctx}} \in \mathbb{R}^{N \times N \times D}$. This matrix can be treated as containing D latent adjacency matrices of the shape $N \times N$, which summarises the global context aware relationship cues between each pair of nodes in a D dimensional space.

Building on the learned $\mathbf{e}^{\text{Ctx}}$, we describe the global contextual relationship cues between $\mathbf{v_i}$ and $\mathbf{v_j}$ (i.e., their edge feature) using the corresponding vector $\mathbf{e}^{\text{Ctx}}_{\mathbf{i,j}} \in \mathbb{R}^{1 \times D}$ in $\mathbf{e}^{\text{Ctx}}$. Then, the final global contextual representation $\mathbf{\hat{e}}^{\mathbf{Ctx}}_{\mathbf{i,j}} \in \mathbb{R}^{1 \times D}$ is obtained by:

$$\mathbf{\hat{e}}^{\mathbf{Ctx}}_{\mathbf{i,j}} = \text{FC}(\mathbf{e}^{\mathbf{Ctx}}_{\mathbf{i,j}}|W_5) \quad \text{for} \quad \forall \mathcal{A}_{i,j} = 1 \tag{3}$$

where $W_5 \in \mathbb{R}^{D \times D}$ is also a learnable matrix. In summary, the GCRM module is learned to generate global context-aware node relationship matrix $\mathbf{\hat{e}}^{\text{Ctx}}$, as each of its vector is produced to describe the global contextual information for a corresponding edge (i.e., the vector $\mathbf{\hat{e}}^{\text{Ctx}}_{\mathbf{i,j}}$ describes the edge connecting nodes $\mathbf{v_i}$ and $\mathbf{v_j}$). Since the $\mathbf{\hat{e}}^{\text{Ctx}}_{\mathbf{i,j}}$ is produced by considering all node features (i.e., the matrix $\mathcal{V}$ that contain all node features), it encodes global context-aware relationship cues between $\mathbf{v_i}$ and $\mathbf{v_j}$, and allows their message passing mechanism also to be global context-aware.

2.3. Fusion of local and global representations

Subsequently, we combine: (i) local relationship representation $\mathbf{e}^{\text{Local}}_{\mathbf{i,j}}$; (ii) global contextual relationship representation $\mathbf{\hat{e}}^{\text{Ctx}}_{\mathbf{i,j}}$; and (iii) the initial input edge feature, as the final multi-dimensional edge representation $\mathbf{\hat{e}_{i,j}}$ to describe task-specific and global context-aware relationships between $\mathbf{v_i}$ and $\mathbf{v_j}$:

$$\mathbf{\hat{e}_{i,j}} = \text{ReLU}(\text{BN}(\mathbf{e}^{\text{Local}}_{\mathbf{i,j}} \oplus \mathbf{\hat{e}}^{\text{Ctx}}_{\mathbf{i,j}} \oplus \text{FC}(\mathbf{e_{i,j}}|W_0))) \tag{4}$$

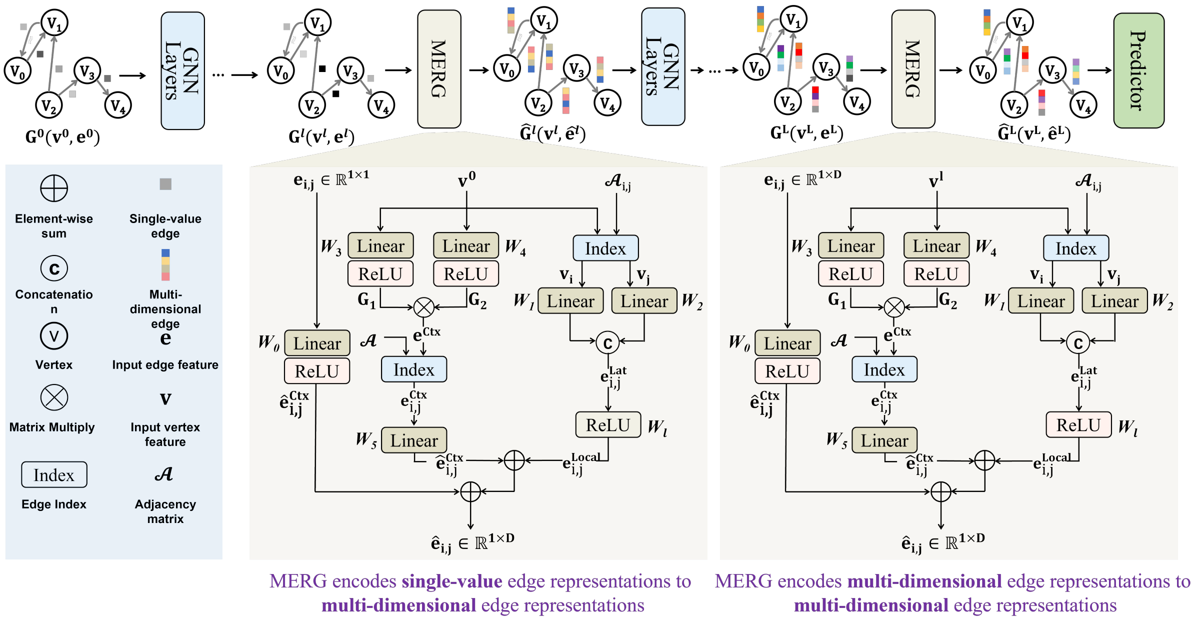

Fig. 1. Illustration of the proposed MERG layer, which can be directly inserted into any GNN. MERG can produce a graph that has multi-dimensional edge features from the input graph with either single-value edges or multi-dimensional edges.

where $\oplus$ denotes the element-wise sum; BN represents the batch normalization; ReLU is the rectified linear activation function. The W_0 is a identity mapping matrix when $\mathbf{e_{i,j}} \in \mathbb{R}^{1\times D}$ and is a learnable vector of the shape $1 \times D$ when $\mathbf{e_{i,j}} \in \mathbb{R}^{1\times 1}$. Subsequently, $\hat{\mathbf{e}}_{\mathbf{i,j}} \in \mathbb{R}^{1\times D}$ can be explicitly represented as $\hat{\mathbf{e}}_{\mathbf{i,j}} = [e_{i,j}(1), e_{i,j}(2), \cdots, e_{i,j}(D)]$.

This way, the proposed MERG **addresses Problem 1** by replacing the single-value edge features with task-specific and global context-aware multi-dimensional edge features for the target input graph, i.e., the relationship between a pair of nodes are described by multiple relationship cues and the message exchanging among nodes are controlled by these multi-dimensional edges during the propagation. In short, our MERG re-formulates the message learned from adjacent nodes during the propagation as:

$$\begin{aligned} \mathbf{m}_{\mathcal{N}(\mathbf{v_i})} =& M(\|_{j=1}^{N} f(\mathbf{v_j}, \hat{\mathbf{e}}_{\mathbf{i,j}})) \\ =& M(\|_{j=1}^{N} f([v_j(1), v_j(2), \cdots, v_j(K)], \\ & [e_{i,j}(1), e_{i,j}(2), \cdots, e_{i,j}(D)])) \end{aligned} \tag{5}$$

where the impact of adjacent nodes is controlled by multiple (D) edge attributes rather than a single value, i.e., richer and more crucial messages from adjacent nodes can be used for each node's updating. Meanwhile, **the Problem 2 is also addressed** as the multi-dimensional edge representation $\mathbf{e}^{\mathbf{l}}_{\mathbf{i,j}}$ can be learned based on only node features, regardless of the initial edge feature $\mathbf{e}^{\mathbf{l-1}}_{\mathbf{i,j}}$'s dimensionality of the input graph.

More importantly, although the number of nodes N and graph topology $\mathcal{A}$ could be different for heterogeneous graphs, all learnable matrices/vectors (W_1, W_2, W_3, W_4, W_5 and W_l) in the MERG are independent of both N and $\mathcal{A}$. As a result, the MERG **can be directly inserted into GNNs for heterogeneous graph analysis**.

3. EXPERIMENTS

Datasets: Eight graph datasets introduced by [8] are employed in this paper, which cover graph-level (MNIST, CIFAR10, PROTEINS, ENZYMES), node-level (PATTERN, CLUSTER) and link-level (TSP, COLLAB) analysis. The details and metrics of these datasets can be found in [8].

Implementation details: We employ the same data splits, optimizer and training settings provided in [8] for all experiments. We use Adam optimizer with a learning rate decay strategy to train all our models. We use GAT and GatedGCN with 4 GNN layers for experiments on MNIST, CIFAR10, PROTEINS and ENZYMES, and 16 GNN layers for experiments on PATTERN, CLUSTER and TSP datasets. The GatedGCN and GAT used for experiments on the COLLAB dataset contain 3 layers.

3.1. Results and discussion

Comparison with state-of-the-art models: We first compare the proposed approach with existing GNN models in Table 1, where our MERG is inserted into two state-of-the-art GNNs (GatedGCN and GAT). It can be observed that both MERG-GatedGCN and MERG-GAT models show clear advantages over the corresponding GatedGCN and GAT on all eight datasets, with average 3.09% and 5.63% improvements,

Task	Graph classification				Node classification		Link prediction	
Dataset	MNIST	CIFAR10	PROTEINS	ENZYMES	PATTERN	CLUSTER	TSP	COLLAB
Model	Test Acc(%)				Test Acc(%)		Test F_1	Test Hits
GCN* [8]	90.71	56.34	76.10	65.83	71.89	68.50	63.10	50.42
R-GCN* [8]	92.49	57.28	-	-	72.67	69.34	-	-
GIN* [8]	96.49	55.26	74.12	65.33	85.59	64.72	66.00	41.73
GAT* [8]	95.54	64.22	76.28	68.50	78.27	70.59	67.10	51.50
GatedGCN* [8]	97.34	67.31	76.36	65.68	86.51	76.08	80.80	52.64
PNA [20]	97.69	70.35	-	-	86.57	-	-	-
GNAS-MP [21]	98.01	70.10	-	-	**86.85**	74.77	74.20	-
EGT [22]	97.72	67.00	-	-	86.83	77.91	81.00	-
DGN [23]	-	72.84	-	-	86.68	-	-	-
ARGNP [24]	-	73.90	-	-	-	77.35	82.10	-
SAT [25]	-	-	-	-	86.84	77.86	-	-
MERG-GAT	98.20	71.66	**82.10(+5.74)**	**75.30(+6.80)**	81.30	74.89	81.20	53.24
MERG-GatedGCN	**98.50(+0.49)**	**74.75(+0.85)**	81.25	70.33	86.71(-0.14)	**78.64(+0.73)**	**83.60(+1.50)**	**53.70(+1.06)**

Table 1. Comparison with the SOTA GNNs on benchmarking results across six medium-scale graph classification and node/link prediction datasets [8], and on TU benchmarking [26] results across two graph classification datasets. * indicates the results are reported in [8]. The MERG-GatedGCN / MERG-GAT represents that GatedGCN or GAT model contains our MERG layer. The numbers in brackets represent the improvements over previous SOTA systems.

respectively. This indicates that the MERG layer is effective in encoding task-specific graph edge representations, but also can be applied to different GNNs. More importantly, graphs processed by our MERG-GatedGCN achieved the SOTA results on five out of eight datasets, while MERG-GAT outperformed previous SOTA approaches on PROTEINS and ENZYMES datasets with more than 5.7% and 6.8% absolute improvements, respectively. Our MERG also produced promising performance on the PATTERN dataset, with only 0.14% lower accuracy than the SOTA. The primary factor contributing to clear improvements brought by our MERG is that it enables the learning of global context-aware multi-dimensional edge features, which are generated through deep-trained layers that are optimized for the target task, which offer a task-specific message passing mechanism for updating node features, thereby facilitating the extraction of superior representations for downstream tasks.

Ablation studies: Our MERG represent each edge by encoding both local relationship feature and global contextual feature. Table 2 investigates their contributions by using: (i) the local relationship feature alone; (ii) the global contextual feature alone; and (iii) the local and global features together. In comparison to graphs with original manually-defined single-value edges, individually learning local relationship features or global contextual features for each edge leads to a more informative graph for all tasks. This finding reveals that global contextual cues contribute more to graph representation than local relationship features, i.e., using only deep-learned global contextual cues as edge representations results in clear improvements over the original single-value edge graph across all datasets. Importantly, our MERG combines the advantages of both local and global relationship cues, where complementary task-specific cues encoded by the MERG suggest that the combination of local and global relationship features achieves better performances than using either of these alone, regardless of the employed GNN.

Table 2. Results achieved for different edge feature settings on six graph datasets. The numbers in brackets represent the improvements over the original GAT/GatedGCN model.

	Baseline	Local	Global	Global + Local
Architecture	Gated-GCN			
CIFAR10	67.31	68.89(+1.58)	73.02(+5.71)	**74.75(+7.44)**
MNIST	97.34	97.55(+0.21)	98.18(+0.84)	**98.50(+1.16)**
CLUSTER	76.08	76.34(+0.26)	77.83(+0.81)	**78.64(+2.56)**
PATTERN	86.51	86.61(+0.10)	86.46(-0.05)	**86.71(+0.20)**
TSP	80.80	80.92(+0.12)	82.40(+1.60)	**83.60(+2.80)**
COLLAB	52.60	**53.70(+1.10)**	-	-
Architecture	GAT			
CIFAR10	64.22	67.7(+3.48)	70.82(+6.60)	**71.66(+7.44)**
MNIST	95.54	96.15(+0.61)	97.89(+2.35)	**98.20(+2.66)**
CLUSTER	70.59	73.23(+2.64)	73.82(+3.23)	**74.89(+4.30)**
PATTERN	78.27	78.53(+0.26)	81.18(+2.91)	**81.30+(3.03)**
TSP	67.10	72.60(+5.50)	80.30(+13.20)	**81.20(+14.10)**
COLLAB	51.50	**53.24(+1.74)**	-	-

4. CONCLUSION

This paper proposes a plugin MERG layer that can produce graphs with multi-dimensional edge features from single-value edge input graph. Results show that it can learn task-specific and complementary cues for each presented edge from local relationship between nodes and global contexts, clearly enhancing performances of different GNNs on various graph datasets. The main limitation is that it has relatively high computational complexity when processing graphs having a large number of high-dimensional vertices, which will be addressed in the future work.

References

[1] S. Song, Y. Song, C. Luo, Z. Song, S. Kuzucu, X. Jia, Z. Guo, W. Xie, L. Shen, and H. Gunes, "Gratis: Deep learning graph representation with task-specific topology and multi-dimensional edge features," *arXiv preprint arXiv:2211.12482*, 2022. 1

[2] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-second AAAI conference on artificial intelligence*, 2018. 1

[3] G. Li, G. Qian, I. Delgadillo, M. Muller, T. Ali, and G. Bernard, "Sgas: Sequential greedy architecture search," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1620–1630. 1

[4] T. Wu, Y. Lu, Y. Zhu, C. Zhang, M. Wu, Z. Ma, and G. Guo, "Ginet: Graph interaction network for scene parsing," in *European Conference on Computer Vision*. Springer, 2020, pp. 34–51. 1

[5] Y. Ou, L. Mi, and Z. Chen, "Object-relation reasoning graph for action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20133–20142. 1

[6] K. Kersting, Nils M. Kriege, C. Morris, P. Mutzel, and M. Neumann, "Benchmark data sets for graph kernels," 2016. 1

[7] Douglas Brent West et al., *Introduction to graph theory*, vol. 2, Prentice hall Upper Saddle River, 2001. 1

[8] V. P. Dwivedi, C. K. Joshi, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *arXiv preprint arXiv:2003.00982*, 2020. 1, 3, 4

[9] E. Abbe, "Community detection and stochastic block models: recent developments," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6446–6531, 2017. 1

[10] E. Isufi, F. Gama, and A. Ribeiro, "Edgenets: Edge varying graph neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 1

[11] M. Mancini, M. Naeem, Y. Xian, and Z. Akata, "Learning graph embeddings for open world compositional zero-shot learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 1

[12] C. Luo, S. Song, W. Xie, L. Shen, and H. Gunes, "Learning multi-dimensional edge feature-based au relation graph for facial action unit recognition," *arXiv preprint arXiv:2205.01782*, 2022. 1

[13] S. Song, Z. Shao, S. Jaiswal, L. Shen, M. Valstar, and H. Gunes, "Learning person-specific cognition from facial reactions for automatic personality recognition," *IEEE Transactions on Affective Computing*, 2022. 1

[14] Z. Shao, S. Song, S. Jaiswal, L. Shen, M. Valstar, and H. Gunes, "Personality recognition by modelling person-specific cognitive processes using graph representation," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 1

[15] Y. Hou, S. Song, C. Yu, W. Wang, and D. Botteldooren, "Audio event-relational graph representation learning for acoustic scene classification," *IEEE signal processing letters*, 2023. 1

[16] L. Gong and Q. Cheng, "Exploiting edge features for graph neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9211–9219. 1

[17] Y. Yang and D. Li, "Nenn: Incorporate node and edge features in graph neural networks," in *Asian conference on machine learning*. PMLR, 2020, pp. 593–608. 1

[18] J. Jo, J. Baek, S. Lee, D. Kim, M. Kang, and S. Hwang, "Edge representation learning with hypergraphs," in *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 1

[19] Z. Wu, J. Paras, W. Matthew, M. Azalia, G. Joseph E, and S. Ion, "Representing long-range context for graph neural networks with global attention," *Advances in Neural Information Processing Systems*, vol. 34, pp. 13266–13279, 2021. 1

[20] C. Gabriele, C. Luca, B. Dominique, L. Pietro, and V. Petar, "Principal neighbourhood aggregation for graph nets," *arXiv preprint arXiv:2004.05718*, 2020. 4

[21] S. Cai, L. Li, J. Deng, B. Zhang, Z. Zha, L. SU, and Q. Huang, "Rethinking graph neural architecture search from message-passing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 4

[22] M. Hussain and M. Zakiand D. Subramanian, "Edge-augmented graph transformers: Global self-attention is enough for graphs," *arXiv preprint arXiv:2108.03348*, 2021. 4

[23] D. Beaini, S. Passaro, V. Létourneau, W. Hamilton, G. Corso, and P. Liò, "Directional graph networks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 748–758. 4

[24] S. Cai, L. Li, X. Han, J. Luo, Z. Zha, and Q. Huang, "Automatic relation-aware graph network proliferation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 4

[25] D. Chen, O. Leslie, and B. Karsten, "Structure-aware transformer for graph representation learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 3469–3489. 4

[26] Morris, N. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "Tudataset: A collection of benchmark datasets for learning with graphs," *arXiv preprint arXiv:2007.08663*, 2020. 4