Vision-GNN Powered Object Detection

Sagar Prakash Barad [*1] **Rucha Balachandra Joshi** [*2]

Abstract

This work aims to explore the utility of ViG in object detection by utilizing graph-based image representations for predictions. While traditional object detection models rely on Deep CNN backbones for feature extraction, our approach involves using ViG to extract feature vectors directly from the image representation. Our goal is to leverage ViG GNN as an efficient feature extractor and compare its performance with models of similar parameter sizes and computational complexities (FLOPs).

1. Introduction

Traditionally, computer vision tasks have predominantly relied on Deep Convolutional Neural Networks (DeepCNNs) as the backbone for visual tasks. However, the landscape is rapidly evolving with the emergence of transformer models with self-attention mechanisms [4] and Multi-Layer Perceptron (MLP)-based vision models [5] capable of achievinng comparable performance.

The motivation for using convolutional models lies in their treatment of input images as regular grids of pixels in Euclidean space. However, when applying a large number of layers, spatial information can be lost. Instead of adhering to grid or sequence structures like in ViT [4], our goal is to employ a generalized graph representation introduced by Han et al. in their Vision-GNN model [1], or Visual Graph Neural Network (**ViG**), for downstream visual tasks.

ViG transforms an image into patches, treating each patch as a node in a graph. It consists of two main components: the Grapher module and the FFN (feed-forward network) module. The Grapher module utilizes graph convolution techniques for efficient information processing within the graph. To address the issue of over-smoothing [12], the model incorporates an FFN module. This module transforms node features and enhances diversity among nodes.

We aim to evaluate ViG's effectiveness for visual tasks, primarily on object detection. This entails using the graph representation as input for graph convolutions and leveraging the embedded space's representation for predictions. Object detection models typically employ Deep CNN backbones for initial feature extraction and use these representations for predictions. Similarly, we can directly employ ViG for image classification and construct feature vectors from the image representation. Our objective is to utilize ViG GNN as a robust feature extractor and compare it with models of similar parameter size and computational complexity (FLOPs).

2. Literature Review

The evolution of image representation models commenced with Convolutional Neural Networks (CNNs), notably LeNet [6], revolutionizing computer vision. CNNs dominated tasks like image classification and object detection. The next milestone was the introduction of ResNet [9], which addressed the vanishing gradient problem, enabling deep network training and enhancing performance. MobileNet [10] accompanied ResNet, introducing lightweight models for edge and mobile devices, expanding accessibility and edge computing possibilities.

In 2020, a paradigm shift occurred with the advent of vision transformers, exemplified by ViT [4]. These models harnessed self-attention mechanisms and transformer capabilities, departing from the conventional CNN approach. Further enhancements in vision transformers introduced pyramid structures, focused attention mechanisms, and advanced position encoding techniques [11], collectively elevating their performance. Concurrently, Multi-Layer Perceptrons (MLPs) [8] gained traction as CNN alternatives, offering specialized modules that excelled in object detection and segmentation, expanding choices for computer vision practitioners.

Concurrently, Graph Neural Networks (GNNs) emerged as a novel approach for handling complex relational data. Originating from foundational works [13], spatial-based Graph Convolutional Networks (GCNs) were refined by Micheli [14], while spectral-based GCNs, pioneered by Bruna et al. [12], harnessed spectral graph theory for graph convolutions. GCNs found diverse applications, spanning social networks [17], citation networks [16], and biochemical graphs [15]. In the realm of computer vision, GCNs fueled innovation, supporting tasks ranging from 3D point cloud classification to scene graph generation and human action recognition. For instance, GCNs processed point clouds derived from LiDAR scans, streamlining tasks like classification and segmentation [20]. Complex challenges like scene graph generation,

entailing the parsing of images into object and relationship graphs, benefited from the synergy of object detectors and GCNs [19]. In the domain of human action recognition, GCNs were leveraged for processing graphs representing interconnected human joints [18].

However, traditional GCNs had limitations when it came to general computer vision tasks that required direct processing of image data. This highlighted the need for dedicated GCN-based backbone networks tailored to the nuances of image data.

3. Model Architecture

The ViG model features two crucial components: the Grapher module for information aggregation using graph convolution and the FFN module with two linear layers for node feature transformation. This section elaborates on ViG's image-to-graph transformation and efficient information processing for visual tasks.

3.1. Graph Construction

The graph construction involves three phases. Initially, an image of size $H \times W \times 3$ is divided into N user-defined patches. Each patch is processed by a deep CNN model to obtain feature vectors, which are then aggregated to form a feature matrix X with dimension D. These feature vectors represent unordered nodes, constituting the set $\mathcal{V}$ within the graph G. Each node v_i connects to its K nearest neighbors, establishing edges e_{ji} between nodes and resulting in the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

3.2. Graph-level Processing

The feature matrix $X \in \mathbb{R}^{N \times D}$ constructs the graph $\mathcal{G} = G(X)$. Graph Convolutional Networks (GCNs) are applied to exchange information among nodes using learnable weight matrices W_{agg} and W_{update} for node aggregation and feature updates. This graph-level processing is denoted as $X' = \mathrm{GraphConv}(X)$.

$$\mathcal{G}' = F(\mathcal{G}, \mathcal{W}) = \mathrm{Update}\left(\mathrm{Aggregate}\left(\mathcal{G}, W_{\text{agg}}\right), W_{\text{update}}\right), \tag{1}$$

$$\mathbf{x}_i' = h\left(\mathbf{x}_i, g\left(\mathbf{x}_i, \mathcal{N}\left(\mathbf{x}_i\right), W_{\text{agg}}\right), W_{\text{update}}\right), \tag{2}$$

where $\mathcal{N}\left(\mathbf{x}_i^l\right)$ is the set of neighbor nodes of $\mathbf{x}_i^l$. Here we adopt max-relative graph convolution [30] for its simplicity and efficiency:

$$g(\cdot) = \mathbf{x}_i'' = \left[\mathbf{x}_i, \max\left(\{\mathbf{x}_j - \mathbf{x}_i \mid j \in \mathcal{N}\left(\mathbf{x}_i\right)\}\right)\right], \tag{3}$$

$$h(\cdot) = \mathbf{x}_i' = \mathbf{x}_i'' W_{\text{update}}, \tag{4}$$

The above graph-level processing can be denoted as $X' = \mathrm{GraphConv}(X)$.

In graph convolution, we employ a multi-head update operation, dividing the aggregated feature $\mathbf{x}_i''$ into multiple heads, denoted as W^1, W^2, up to W^h. Each head undergoes an independent update with distinct weight matrices, and their results are concatenated to form final node representations:

$$\mathbf{x}i' = \left[\text{ head }{}^1 W\text{update }{}^1,\ \text{head }{}^2 W^2_{\text{update}}, \ldots,\ \text{head }{}^h W^h_{\text{update}}\right]. \tag{5}$$

This multi-head approach enhances feature diversity by allowing each head to focus on different aspects of the input data, resulting in a more comprehensive and expressive representation. It enables the model to capture intricate patterns and relationships in the graph structure, leading to improved performance and flexibility.

3.3. ViG Block

To address over-smoothing in deep GCNs, ViG introduces feature transformations and non-linear activations. Linear layers are added before and after graph convolution operations to align node features and enhance diversity. Non-linear activation functions maintain model expressiveness. At each node, feed-forward networks (FFNs) are introduced to further enhance capacity and mitigate over-smoothing. ViG constructs a ViG block by stacking Grapher and FFN modules, serving as the foundational unit for the network. This ViG network, utilizing graph image representations and ViG blocks, sustains feature diversity in deeper layers, making it suitable for various visual tasks.

$$Y = \sigma\left(\mathrm{GraphConv}\left(XW_{\text{in}}\right)\right) W_{\text{out}} + X, \tag{6}$$

Where $Y \in \mathbb{R}^{N \times D}$ represents the transformed features, and W_{in} and W_{out} are the weights of the fully-connected layers.

$$Z = \sigma\left(YW_1\right) W_2 + Y \tag{7}$$

Here, $Z \in \mathbb{R}^{N \times D}$, where W_1 and W_2 represent the weights of fully-connected layers.

4. Models Built

We've adopted an isotropic architecture for ViG, similar to ViT [4], to ensure a fair comparison. This architecture maintains consistent feature size and shape throughout the network. We've created six ViG variants: three for 32 x

Table 1. Isotropic Architectur ViG on CIFAR Data

Model	Depth	Dimension D	Params (M)	FLOPs (B)
ViG-Ti	12	192	5.5	5.3
ViG-S	16	320	20.1	20.3
ViG-B	16	640	79.6	80.929

Table 2. Isotropic ViG Architecture on ImageNet Data

Model	Depth	Dimension D	Params (M)	FLOPs (B)
ViG-Ti	12	192	7.1	1.3
ViG-S	16	320	22.7	4.5
ViG-B	16	640	86.8	17.7

32 images (for testing) and three for 224 x 224 images. In CIFAR test runs, we've reduced the number of deep CNN layers to 16 nodes (compared to 196 for ImageNet). These models, denoted as ViG-Ti, S, and B, vary in size. Node count is standardized at N = 196. To gradually expand the receptive field, we linearly increase the number of neighboring nodes (K) from 9 to 18 as the network deepens with number of heads to h = 4 by default.

5. Results

In our initial experiments, we utilized the CIFAR dataset, which includes CIFAR-10 and CIFAR-100. CIFAR provides 60,000 32x32 color images, distributed across 10 and 100 classes, respectively. For broader image classification tasks, we turned to the ImageNet ILSVRC 2012 dataset, which offers an extensive collection of 1.2 million training images and 50,000 validation images, grouped into 1000 classes. For access and licensing details for the ImageNet dataset, visit http://www.image-net.org/download.

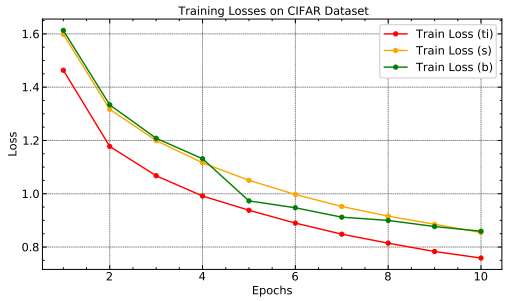

Figure 1. The training process involved 10 epochs of training with the AdamW optimizer, a batch size of 1024, an initial learning rate of 2e-3, a cosine learning rate schedule with a weight decay of 0.05.

In our training runs, we've observed promising results, and with more training epochs, it's likely that we can achieve similar or even better accuracy on the ImageNet dataset compared to models like ResNet variants. We're directly comparing our ViG-Ti, ViG-S, and ViG-B models to ResNet-18, ResNet-50, and ResNet-101, which are commonly used as backbones in various models. Our ViG models are comparable in size to these ResNet models, both in terms of FLOPs and parameters. According to the paper's findings, ViG has shown better performance in image classification compared to ResNets. These results suggest that using a graph representation of images might be superior to traditional CNNs.

Table 3. ViG models on CIFAR Dataset

Model	Top-1	Top-5
ViG-Ti	66.1	97.67
ViG-S	65.64	97.95
ViG-B	67.71	98.78

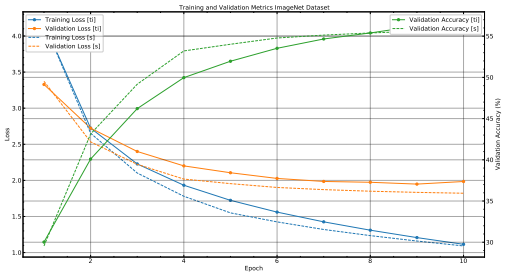

Figure 2. The training process involved 10 epochs of training with the AdamW optimizer, a batch size of 256 for the Tiny variant and 128 for the Small variant, an initial learning rate of 2e-3, a cosine learning rate schedule, and a weight decay of 0.05.

6. Conclusion

This work applies Vision GNN to process images as graphs using GCNs, enhancing complex object handling in deep learning. To address over-smoothing, we introduce node-level feature transformations for increased information diversity.Current experiments have shown promising results, indicating that with further training, our models will possess comparable or even superior representation capabilities to models like ResNets.In the future, we aim to train the ViG on COCO Dataset, PubTables 1M Dataset, and FinTab Dataset. Moreover, we aim to implement LT GNN to enhance transfer learning capabilities..

Table 4. ViG models on ImageNet Dataset

Model	Top-1	Top-5
ViG-Ti	55.60	93.32
ViG-S	55.75	93.95

References

[1] Han, K., Wang, Y., Guo, J., Tang, Y., & Wu, E. *Vision GNN: An Image is Worth Graph of Nodes.* arXiv preprint arXiv:2206.00272v3, 2022.

[2] He K, Zhang X, Ren S, Sun J. *Deep residual learning for image recognition. CVPR.* 770-778, 2016.

[3] Krizhevsky A, Sutskever I, Hinton GE. Imagenet *classification with deep convolutional neural networks. NeurIPS.* 1097-1105, 2012.

[4] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, et al. *An image is worth 16x16 words: Transformers for image recognition at scale.* ICLR. 2021.

[5] Touvron H, Bojanowski P, Caron M, Cord M, El-Nouby A, Grave E, Izacard G, Joulin A, Synnaeve G, Verbeek J, et al. Resmlp: *Feedforward networks for image classification with data-efficient training.* arXiv preprint arXiv:2105.03404. 2021.

[6] LeCun Y, Bottou L, Bengio Y, Haffner P. *Gradient-based learning applied to document recognition.* Proceedings of the IEEE. 86(11):2278–2324, 1998.

[7] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. *Swin transformer: Hierarchical vision transformer using shifted windows.* In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 10012–10022, 2021.

[8] Chen, S., Xie, E., Ge, C., Liang, D., & Luo, P. *CycleMLP: A MLP-like architecture for dense prediction.* In International Conference on Learning Representations (ICLR). (2022).

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep residual learning for image recognition.* In CVPR, pages 770–778, 2016

[10] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. *Mobilenets: Efficient convolutional neural networks for mobile vision applications.* arXiv preprint arXiv:1704.04861, 2017.

[11] Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and improving relative position encoding for vision transformer. In ICCV, pages 10033–10041, 2021.

[12] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203, 2013.

[13] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In IJCNN, volume 2, pages 729–734, 2005.

[14] Alessio Micheli. Neural network for graphs: A contextual constructive approach. IEEE Transactions on Neural Networks, 20(3):498–511, 2009.

[15] Nikil Wale, Ian A Watson, and George Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. Knowledge and Information Systems, 14(3):347–375, 2008.

[16] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. AI magazine, 29(3):93–93, 2008.

[17] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In NIPS, pages 1025–1035, 2017.

[18] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeletonbased action recognition. In AAAI, 2018.

[19] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In ECCV, pages 670–685, 2018.

[20] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In CVPR, pages 7074–7083, 2020.