DVHGNN: Multi-Scale Dilated Vision HGNN for Efficient Vision Recognition

Caoshuo Li[1,2*], Tanzhe Li[1,2*], Xiaobin Hu[3], Donghao Luo[3], Taisong Jin[1,2†]
[1]Key Laboratory of Multimedia Trusted Perception and Efficient Computing,
Ministry of Education of China, Xiamen University, China.
[2]School of Informatics, Xiamen University, China. [3]Tencent Youtu Lab.
{licaoshuo, tanzheli}@stu.xmu.edu.cn, jintaisong@xmu.edu.cn,
{xiaobinhu, michaelluo}@tencent.com

Abstract

Recently, Vision Graph Neural Network (ViG) has gained considerable attention in computer vision. Despite its groundbreaking innovation, Vision Graph Neural Network encounters key issues including the quadratic computational complexity caused by its K-Nearest Neighbor (KNN) graph construction and the limitation of pairwise relations of normal graphs. To address the aforementioned challenges, we propose a novel vision architecture, termed ***D****ilated* ***V****ision* ***H****yper****G****raph* ***N****eural* ***N****etwork (DVHGNN), which is designed to leverage multi-scale hypergraph to* efficiently *capture high-order correlations among objects. Specifically, the proposed method tailors Clustering and* ***D****ilated* ***H****yper****G****raph* ***C****onstruction (DHGC) to adaptively capture multi-scale dependencies among the data samples. Furthermore, a dynamic hypergraph convolution mechanism is proposed to facilitate adaptive feature exchange and fusion at the hypergraph level. Extensive qualitative and quantitative evaluations of the benchmark image datasets demonstrate that the proposed DVHGNN significantly outperforms the state-of-the-art vision backbones. For instance, our DVHGNN-S achieves an impressive top-1 accuracy of* ***83.1%*** *on ImageNet-1K, surpassing ViG-S by* ***+1.0***↑ *and ViHGNN-S by* ***+0.6***↑.

1. Introduction

The rapid advancement of deep learning has significantly propelled computer vision community. Convolutional Neural Networks (CNNs) [6, 17, 19, 26, 42, 43, 45] have become the predominant approach in various vision tasks, efficiently capturing the spatial relationships and structural complexities within images owing to the locality and shared weights. However, CNNs are constrained by their narrow focus on local information, rendering them incapable of capturing the long-range dependencies. The introduction of the Transformer [47], renowned for handling long-range information in natural language processing (NLP), to computer vision via the Vision Transformer (ViT) [9] marked a significant shift. ViT demonstrates strong performance across various vision tasks by efficiently modeling long-range dependencies of an image. Despite its strengths, ViT still has the drawbacks including the absence of CNNs' inductive bias, a dependency on extensive training data and the quadratic computational complexity caused by its global self-attention mechanism. Subsequently, various variants and extensions [28, 32, 37, 46, 49, 53] of ViT have been proposed to mitigate these drawbacks, offering more efficient and adaptable solutions for different vision tasks.

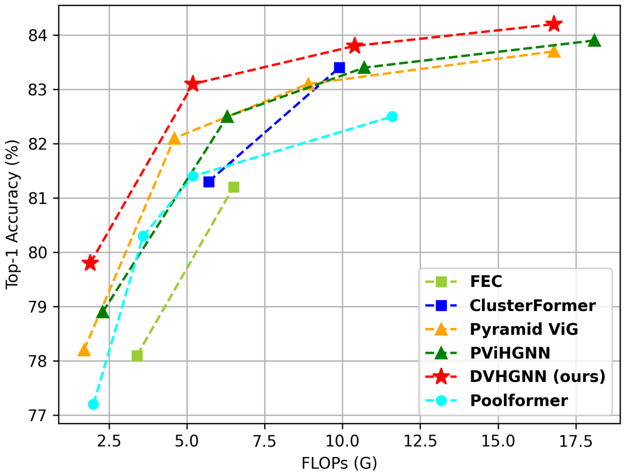


Figure 1. Comparison of FLOPs and Top-1 accuracy on ImageNet-1K. The proposed DVHGNN achieves the best performance compared to other state-of-the-art models.

Both CNNs and ViTs treat images as grid and sequential structures, respectively, lacking the flexibility and the ability to capture the structure of complex objects in an image.

[*] Equal contribution [†] Corresponding author

Recently, Vision Graph neural network (ViG) [15] transforms images into graph structures within non-Euclidean space and leverages Graph Neural Networks (GNNs) to employ feature exchange and fusion at the graph level. This marks the first successful generalization of GNNs to a general vision backbone, bridging the gap in capturing complex object structures with enhanced flexibility.

However, ViG still faces two key limitations. *On the one hand*, despite efforts to extract deeper semantic information through an increased number of blocks, ViG fails to represent the complex inter-class and intra-class relationships among objects in an image effectively. This limitation stems from its normal graph structure, designed to modeling the connections of low-level and local features between neighboring nodes that fails to capture the high-order correlations among more than two nodes. *On the other hand*, the employment of the KNN graph in ViG has quadratic computational complexity, which requires a substantial amount of memory and computational resources. Furthermore, the inherent non-learnability of the KNN graph strategy results in the potential information lost in the graph construction.

To overcome the abovementioned issues, models like ViHGNN [16] are proposed to enhance ViG's capability via hypergraph capturing high-order correlations among objects. However, ViHGNN shows marginal performance improvement based on two reasons:

1) Limitations of Hypergraph Construction: ViHGNN employs the Fuzzy C-means algorithm to construct hypergraphs across the entire image, neglecting the significance of local and multi-scale features, resulting in quadratic computational complexity for the number of hyperedges.

2) Reciprocal Feedback Limitations: Although ViHGNN introduces a reciprocal feedback mechanism between patch embeddings and the hypergraph for message passing and mutual optimization, the nature of the Fuzzy C-means still constrains the model's adaptability during the learning process. Specifically, ViHGNN lacks the capability to effectively model the dynamic connections between hypergraph structure and hypergraph convolution.

To address the issues above, we propose a novel Vision Hypergraph Neural Network, termed **D**ilated **V**ision **H**yper**G**raph **N**eural **N**etwork (DVHGNN), which introduces a multi-scale hypergraph approach to image representation. Specifically, a multi-scale hypergraph is constructed using cosine similarity clustering and Dilated HyperGraph Construction (DHGC) techniques. In this process, each hyperedge generates a centroid, allowing each vertex to adaptively perform dynamic hypergraph convolution by utilizing either cosine similarity or sparsity-aware weights relative to the centroid. Our contributions are summarized as follows:

- We propose a novel paradigm termed DVHGNN, which is designed to leverage multi-scale hypergraph to efficiently capture high-order correlations among objects.
- To obtain multi-scale hypergraph representations of the images, we adopted a dual-path design that includes clustering and DHGC, simultaneously focusing on local and sparse multi-scale information.
- To better facilitate information exchange at the hypergraph level, we propose a novel two-stage Dynamic Hypergraph Convolution framework, which leverages the pairwise cosine similarity and sparsity-aware weight of hyperedges to adaptively aggregate vertex embeddings into hyperedge features to update original embeddings.
- Extensive experiments demonstrate the superior performance of our DVHGNN. Specifically, our DVHGNN-S achieves an impressive Top-1 accuracy of 83.1% on ImageNet-1K, surpassing ViG-S by 1.4% and ViHGNN-S by 0.6% with fewer parameters and FLOPs.

2. Realted Works

2.1. CNNs and Transformer for Vision

Convolutional Neural Networks (CNNs) [27] have been the cornerstone of computer vision, since AlexNet [26] marked a significant breakthrough, leading to the development of various influential CNN architectures including VGG [42], GoogleNet [43], ResNet [17], and MobileNet [19]. Inspired by the success of Transformer [47] in natural language processing, Vision Transformer [9] was introduced into vision tasks, which is used as the basis of the subsequent models like DeiT [46], PVT [49], and Swin-Transformer [32] to achieve impressive performances across various downstream vision tasks. Furthermore, hybrid architectures such as CvT [52], Coatnet [7], and ViTAE [57] have been proposed to amalgamate the strengths of convolutions and Transformers. Recently, larger kernel CNNs, such as ConvNeXt [33] and RepLKNet [8], have demonstrated strong competitive capabilities, highlighting the diverse and evolving landscape in advancing computer vision technologies. However, CNNs still struggle with long-range modeling.

2.2. Graph/Hypergraph Neural Network

Graph Neural Networks (GNNs) [12, 40] were initially used for processing non-Euclidean data and have achieved various applications in social networks [13], citation networks [41], and biochemical graphs [48]. Currently, GNNs have been applied to computer vision, including object detection [55], and point cloud classification [58].

In computer vision, HGNNs are regarded as a significant extension of GNNs, distinguished by their ability to capture high-order relationships through hypergraphs. This shift enhances the understanding and representation of complex data structures, as demonstrated in applications such as image retrieval [23], 3D object classification [11, 22], and person re-identification [66]. Specifically, HGNNs improve image retrieval by modeling images as vertices con-

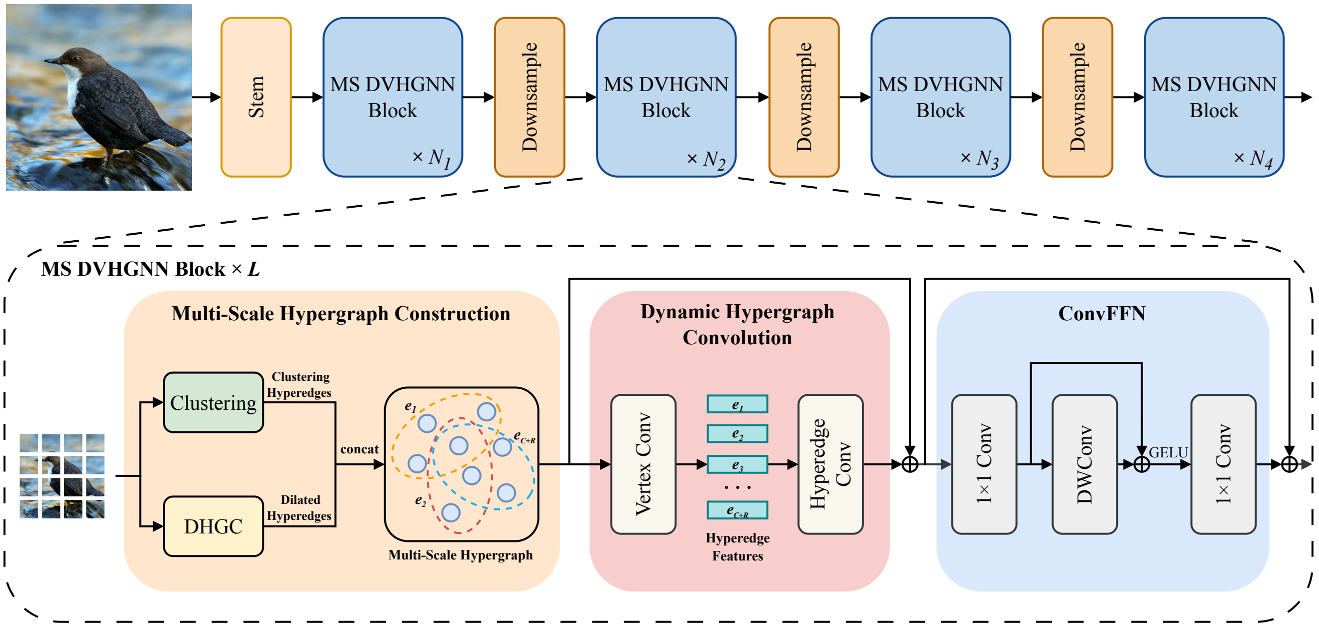


Figure 2. Architecture of the proposed DVHGNN. In each block, Multi-Scale(MS) DVHGNN block constructs multi-scale hyperedges, followed by message passing through vertex and hyperedge convolutions, and finalizes with ConvFFN to enhance feature transformation capacity and counteract over-smoothing.

nected by hyperedges based on feature correlations, thereby refining the retrieval process with greater sensitivity to image relationships. Similarly, in 3D object classification, HGNNs enhance accuracy by leveraging vertices for object representation and hyperedges to capture the complex relationships between different views. Despite the contributions made by GNNs and HGNNs, there has been limited use of GNNs or HGNNs as the backbone in computer vision. ViG [15] and ViHGNN [16] are among the few works that treat patches within input images as nodes and construct graphs/hypergraphs through different strategies like KNN or Fuzzy C-means. However, both KNN and Fuzzy C-means face challenges in high-dimensional spaces, suffer from computational inefficiency, and struggle to capture complex patterns, making them less suitable for modern visual tasks compared to more adaptive deep learning models.

2.3. Graph/Hypergraph Structure Learning

In GNNs and HGNNs, the performance is significantly influenced by the quality of graph and hypergraph structures. Recent advances have focused on structure learning to enhance the effectiveness of these models. For GNNs, methods such as those in [4, 10] have been developed to jointly learn graph structures and node embeddings, relying on graph adjacency matrix properties like low-rank and sparsity. Other approaches [35, 67] dynamically adjust graph structures by learning metrics or distributions for the edges.

Hypergraph structure learning, while less explored, has seen advances with models like DHSL [64, 65] applying dual optimization for simultaneous learning of label projection matrices and hypergraph structures. DHGNN [24] uses K-Means and KNN for adaptive hypergraph construction, and HERALD [63] focuses on optimizing hypergraph Laplacian matrices. Challenges include non-convex optimization issues and high computational demands, especially noted in DHSL. Expanding the repertoire of structure learning, recent works include HyperSAGE [1] for scalable hypergraph learning and Dynamic HyperEdge Convolution Networks (DHECN) that adjust hyperedge weights dynamically. However, these methods are constrained by their graph construction techniques, preventing them from adaptively adjusting their hypergraph structures.

3. Methods

In this section, we begin by revisiting the concept of hypergraph. Subsequently, we delineate the process of the proposed multi-scale hypergraph image representation. Finally, we elaborate on the design of the dynamic hypergraph convolution and the multi-head computation mechanism.

3.1. Preliminary:Recap the Concept of Hypergraph

Notations. For an image with dimensions $H \times W \times 3$, we employ Vision GNN [15] to divide it into N patches. A linear layer is then applied to the derived patches to convert them into high-dimensional vectors $\mathbf{x}_i \in \mathbb{R}^D$. This transformation results in a matrix $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$,

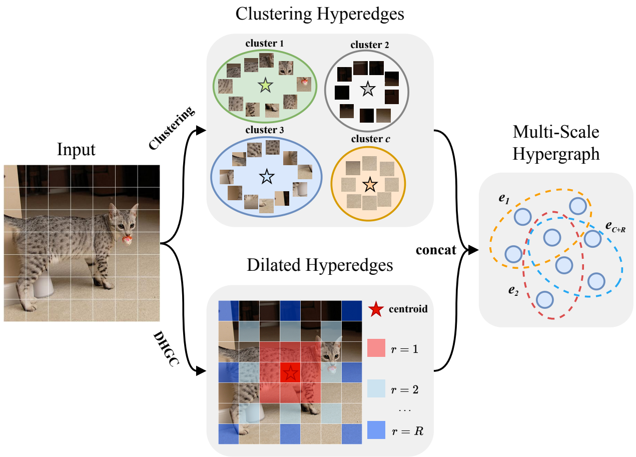


Figure 3. Illustration of Multi-Scale Hypergraph Construction (without region partition). The final hyperedge set is composed of two types of hyperedges: a set of size C obtained from cosine similarity clustering, and a set of size R derived from DHGC. Each hyperedge corresponds to a hyperedge centroid, marked with a pentagon in the diagram. By default, R = 3, with the distinct dilated hyperedges corresponding to a kernel size of 3 × 3 with dilation rates r = 1, 2, and 3, respectively, resulting in receptive field sizes of 3 × 3 , 5 × 5 , and 7 × 7 .

where D represents the dimensionality of the features and i indexes the patches with $i = 1, 2, \ldots, N$. These vectors can be conceptualized as a collection of unordered vertices, denoted by $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$. A hypergraph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of hyperedges. The set of incident edges of vertex i is denoted by E_i, where $E_i = \{e \in E \mid v_i \in e\}$. The weight of each hyperedge is encoded by $\mathbf{W}$, a diagonal matrix of edge weights. Differing from a normal graph, each hyperedge can connect any number of vertices.

3.2. Multi-Scale HGNN Representation of Image

In this subsection, we propose the Cluster and Dilated Hypergraph Construction (DHGC) based Hypergraph representation. Our hypergraph construction approach includes two distinctive hyperedge types: ones derived from clustering and the others obtained via DHGC. Initially, a hyperedge set is generated through a clustering technique. The derived set is then augmented by employing a DHGC mechanism, culminating in a multi-scale hyperedge collection characterized by its sparsity. The details of our Multi-Scale Hypergraph Construction are shown in Figure. 3.

Clustering. Given a set of vertices $\mathcal{V}$ and associated feature vectors $\mathbf{X} \in \mathbb{R}^{N \times D}$, vertices are grouped into the distinct clusters reflecting their mutual similarities and ensuring that each vertex belongs exclusively to a single cluster. Initially, the feature vectors $\mathbf{X}$ are mapped onto a similarity space $\mathbf{X}_s$. In this space, C centroids $\mathbf{X}_c$ are established uniformly, and their characteristics are distilled using an average pooling methodology. Subsequently, the cosine similarity matrix $\mathbf{S}$ between $\mathbf{X}_s$ and $\mathbf{X}_c$ is computed, resulting in $\mathbf{S} \in \mathbb{R}^{C \times N}$. Based on the similarity matrix $\mathbf{S}$, each vertex is then assigned to the nearest hyperedge. This process yields a primary set of hyperedges, denoted as $\mathcal{E}_c$.

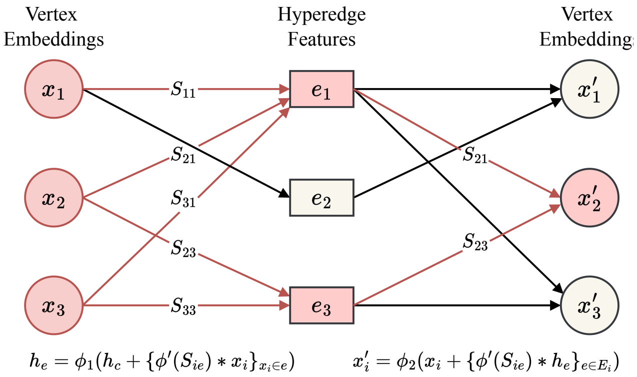


Figure 4. Illustration of two-stage message passing of our Dynamic Hypergrpah Convolution (DHConv). $\boldsymbol{h}_c$ is the feature of the hyperedge centroid, $\mathbf{S}_{ie}$ is the cosine similarity matrix between vertices and hyperedge centroids, and $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$ represent the vertex feature before and after DHConv. Note that how messages flow to vertex 2 is marked in red.

Region Partition. In terms of computational efficiency, the time complexity for comparing N D-dimensional vectors across C clusters is $O(NCD)$. To mitigate excessive computational demands, the image is partitioned into m subregions akin to the Swin Transformer [32]. This strategy reduces the time complexity to $O\left(\frac{NCD}{m}\right)$. While it may limit the scope of global information exchange, it introduces a beneficial inductive bias by promoting locality, akin to the sliding window approach observed in CNNs.

Dilated Hypergraph Construction. We propose a novel hypergraph construction strategy, named Dilated Hypergraph Construction (DHGC), to enhance the hyperedge set derived from clustering processing. In this approach, for each $w \times w$ window, the central vertex v_c forms a series of dilated hyperedges. For the central vertex v_c in each window, we define a set of hyperedge neighborhoods $N_k(v_c)$ for different dilation rates $r \in \mathbb{N}^+$. These hyperedge neighborhoods connect the central vertex in the window to the vertices in its dilated neighborhood.

For a given dilation rate r, the hyperedge neighborhood $N_k(v_c)$ for each $w \times w$ window's central vertex v_c includes K vertices, identified within a dilation step r. The hyperedge comprises vertices whose coordinates, after dilation, remain inside the window's boundary.

The coordinates are defined as follows:

$$\begin{aligned} N_k(v_c) = \{(i', j') | i' = i + p \times r, j' = j + q \times r\}\,, \\ -\frac{w}{2r} \leq p, q \leq \frac{w}{2r}, |N_k(v_c)| = K, \end{aligned} \tag{1}$$

where (i, j) are the coordinates of v_c, p, q are integers that scale with the dilation rate r, and $|N_k(v_c)|$ represents the number of vertices within the hyperedge structure. This arrangement ensures that each hyperedge captures a fixed number of vertices, enabling a consistent representation of spatial relationships across different scales.

Each dilated hyperedge is assigned to a unique sparsity-aware weight $\mathbf{w}_r$, reflecting its specific dilation rate's contribution to the feature aggregation process. If there are R dilation rates considered, the final weight matrix $\mathbf{W}$ for the hypergraph is obtained by combining the R individual hyperedge weights $\mathbf{w}_r$, leading to:

$$\mathbf{W} = \text{diag}(\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_R). \tag{2}$$

Consequently, our *Dilated Hypergraph* is mathematically formulated as follows:

$$\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W}, \mathbf{S}) = \left\{ \mathcal{V}, \bigcup_{v_c \in \mathcal{V}} \mathcal{E}_{k,v_c} \cup \mathcal{E}_c, \mathbf{W}, \mathbf{S} \right\}, \tag{3}$$

where $\mathcal{V}$ represents the entire set of vertices within the image or dataset, and $\mathcal{E}$ is the cumulative union of all hyperedge neighborhoods derived from different dilation levels and all hyperedge sets formed by clustering, corresponding to every central vertex v_c within the windows. This structure not only enables the capture of the local information of an image but also facilitates the grasping of neighbor information at extended distances, according to varying dilation levels. By amalgamating features from different levels of dilation, we gain a deeper understanding of the intricate structure and interactions within an image.

3.3. Dynamic Hypergraph Convolution

In this subsection, we detail the architecture of our proposed Dynamic Hypergraph Convolution (DHConv), divided into two principal stages similar to [21, 24]. The first stage, Vertex Convolution, aggregates vertex embeddings to constitute the hyperedge features. The second stage, Hyperedge Convolution, involves the distribution of these hyperedge features back to the associated vertices, which serves to update their individual embeddings.

Vertex Convolution. In previous studies, vertex convolutions have relied on techniques such as pooling, applying pre-defined transformation matrices based on graph structures, and dynamically learning the transformation matrix $\mathbf{T}$ utilizing MLPs and $1-d$ convolutions [24]. Due to the computational complexity when dealing with vision tasks, the existing methods are unsuitable to handle the hypergraph-based vision data.

Our Dynamic Vertex Convolution approach is specifically designed to handle the high dimensionality and complexity of visual data. Unlike conventional methods, we leverage cosine similarity clustering hyperedges $\mathcal{E}_c$, aggregating features adaptively through dynamic learning.

Each given clustering hyperedge e consists of a hyperedge centroid h_c and multiple vertices $x_i \in e$. Our vertex feature aggregation process is formulated as follows:

$$\begin{aligned} h_e &= \frac{1}{C}\left(h_c + \sum_{x_i \in e} \text{sig}(\alpha s_i + \beta) * x_i \right), \\ \text{s.t.} \quad C &= 1 + \sum_{x_i \in e} \text{sig}(\alpha s_i + \beta), \end{aligned} \tag{4}$$

where s_i represents the cosine similarity between vertex i and the hyperedge centroid h_c, x_i are the feature vectors of vertex, and C is a normalization factor in ensuring numerical stability and emphasizing the locality in each hyperedge.

For each given dilated hyperedge e, the feature aggregation process is designed to capture multi-scale information efficiently. This process is formulated as follows:

$$h_e = \frac{h_c + \sum_{x_i \in e} w_r * x_i}{1 + |e| w_r}, \tag{5}$$

where w_r is a learnable parameter, assigned to the dilated hyperedge e with dilation r, reflecting the importance of features at different scales.

Through this approach, the proposed vertex convolution not only adapts to the complexity of hypergraph structure but also dynamically adjusts the interaction between vertices based on their cosine similarity to the hyperedge center. Thus, the proposed method can derive a dynamic hypergraph image representation, which is more suitable for handling highly complex relationships in different vision tasks.

Hyperedge Convolution. The proposed Hyperedge Convolution, inspired by the Graph Isomorphism Network (GIN) convolution [56], adaptively leverages hyperedge features to update the vertex embeddings based on cosine similarity or hyperedge weights, enabling feature exchange and fusion of local vertex information with hyperedge context. The update formulation is defined as follows:

$$z_i = \sum_{e \in E_i} \left(I_c(e) * \text{sig}(\alpha s_i + \beta) + I_d(e) * w_r \right) * h_e, \tag{6}$$

$$x_i' = FC\left(\sigma\left(\text{Conv}((1+\varepsilon)x_i + z_i)\right)\right), \tag{7}$$

where h_e denotes the aggregated feature of hyperedge e, E_i denotes a set of all hyperedges connected to vertex i, ε is a learnable parameter, $\sigma(\cdot)$ is a non-linear activation function, FC is a fully connected layer, $I_c(e)$ and $I_d(e)$ are the indicator functions for the two types of hyperedges, with $I_c(e), I_d(e) \in \{0, 1\}$ and $I_c(e) \neq I_d(e)$.

3.4. Multi-head Computation

Multi-head computations have been extensively studied in transformer architecture [47]. Our DVHGNN model continues to implement these computations and update mechanisms as described in [36]. Specifically, we employ h

Model	Type	Params (M)	FLOPs (G)	Top-1 (%)
ResNet-18 [17]	CNN	12.0	1.8	70.6
ResNet-50 [17]	CNN	25.6	4.1	79.8
InceptionNeXt-T [60]	CNN	28.0	4.2	82.3
InceptionNeXt-S [60]	CNN	49.0	8.4	83.5
InceptionNeXt-B [60]	CNN	87.0	14.9	84.0
Swin-T [32]	ViT	29.0	4.5	81.3
Swin-S [32]	ViT	50.0	8.7	83.0
Swin-B [32]	ViT	88.0	15.4	83.5
PVTv2-B1 [50]	ViT	13.1	2.1	78.7
PVTv2-B2 [50]	ViT	25.4	4.0	82.0
FLatten-Swin-T [14]	ViT	29.0	4.5	82.5
FLatten-Swin-S [14]	ViT	51.0	8.7	83.5
FLatten-Swin-B [14]	ViT	89.0	15.4	83.8
Poolformer-S12 [59]	Pool	12.0	2.0	77.2
Poolformer-S36 [59]	Pool	31.0	11.2	80.3
Poolformer-M48 [59]	Pool	73.0	15.9	81.4
Vim-Tiny [70]	SSM	7.0	-	76.1
Vim-Small [70]	SSM	26.0	5.1	80.5
Vim-Base [70]	SSM	98.0	-	81.9
VMamba-T [31]	SSM	22.0	5.6	82.2
VMamba-M [31]	SSM	44.0	11.2	83.5
VMamba-B [31]	SSM	75.0	18.0	83.7
CoC-S [36]	Cluster	14.0	2.6	77.5
CoC-M [36]	Cluster	27.9	5.5	81.0
ClusterFormer-T [28]	Cluster	28.0	5.7	81.5
ClusterFormer-S [28]	Cluster	48.7	9.9	83.4
FEC-Small [2]	Cluster	5.5	1.4	72.7
FEC-Base [2]	Cluster	14.4	3.4	78.1
FEC-Large [2]	Cluster	28.3	6.5	81.2
Pyramid ViG-Ti [15]	GNN	10.7	1.7	78.2
Pyramid ViG-S [15]	GNN	27.3	4.6	82.1
Pyramid ViG-M [15]	GNN	52.4	8.9	83.1
Pyramid ViG-B [15]	GNN	92.6	16.8	83.7
PViHGNN-Ti [16]	HGNN	12.3	2.3	78.9
PViHGNN-S [16]	HGNN	28.5	6.3	82.5
PViHGNN-M [16]	HGNN	52.4	10.7	83.4
PViHGNN-B [16]	HGNN	94.4	18.1	83.9
DVHGNN-T (ours)	HGNN	11.1	1.9	**79.8**
DVHGNN-S (ours)	HGNN	30.2	5.2	**83.1**
DVHGNN-M (ours)	HGNN	52.5	10.4	**83.8**
DVHGNN-B (ours)	HGNN	92.8	16.8	**84.2**

Table 1. Results of DVHGNN variants and other backbones on ImageNet-1K. All the models are trained at 224 × 224 resolution

heads, standardizing the dimensions of the value space X_v and similarity space X_s to D' for simplicity. The outputs of the multi-head operations are combined and integrated via a FC layer, resulting in h distinct hypergraphs. This multi-head computation allows the model to concurrently update features across multiple representational hypergraph subspaces, thereby enhancing feature diversity.

3.5. Dilated Vision HGNN Architecture

As shown in Figure. 2, the basic component of our proposed architecture is the Multi-Scale (MS) DVHGNN block. Each MS DVHGNN block consists of three modules: MS Hypergraph Construction, DHConv, and ConvFFN. To make the model compatible with various downstream vision tasks, we adopt a four-stage pyramid structure. Within each stage, a convolutional layer is employed to reduce the dimensions of the input feature map to a quarter of its original height and width ($\frac{H}{4} \times \frac{W}{4}$), which is subsequently succeeded by a series of MS ViHGNN blocks. Culminating the architecture, a prediction head is deployed for image classification. Detailed configurations of our DVHGNN variants are listed in appendices, with D' refers to the dimension of each head.

4. Experiments

4.1. Image Classification on ImageNet

Experimental Settings: We benchmark our models on ImageNet-1K [39], a dataset with 1.3 million training images and 50,000 validation images across 1,000 classes. The training was conducted over 300 epochs with an image resolution of 224×224, using AdamW [34] optimizer. We enhanced our models' generalization with a combination of data augmentation and regularization techniques, including RandAugment [5], Mixup [62], CutMix [61], and Random Erasing [68], with weight decay, Label Smoothing [44] and Stochastic Depth [20]. Our implementation leverages PyTorch [38] and Timm [51].

Experimental Results: As depicted in Table. 1, DVHGNN models outperform other state-of-the-art vision backbones, especially the GNN-based and HGNN-based models. Specifically, DVHGNN-S achieves a Top-1 accuracy of 83.1%, surpassing ViG-S by 1.0% and ViHGNN by 0.6% with 18% less FLOPs. These results highlight our model's ability to capture complex visual representations effectively with increased computational efficiency.

4.2. Object Detection and Instance Segmentation

Experimental Settings: The experiments were conducted on the MS-COCO [29] dataset, which includes 118K training images, validation images, and 20K test images. Respectively, for objection detection and instance segmentation, We employed DVHGNN-S pre-trained on ImageNet-1K as a backbone and incorporated it into two detectors: RetinaNet [30] and Mask R-CNN [18]. Following common practice, we trained the two downstream vision tasks with DVHGNN-S for 12 (1 × schedule) epochs and the implementation was performed by MMDetection [3].

Experimental Results: As depicted in Table. 2, our DVHGNN outperforms the state-of-the-art back-

Backbone	RetinaNet 1×							
	Param	FLOPs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50 [17]	38M	239G	36.3	55.3	38.6	19.3	40.0	48.8
PVT-Small [49]	34M	227G	40.4	61.3	44.2	25.0	42.9	55.7
Swin-T [32]	39M	245G	41.5	62.1	44.2	25.1	44.9	55.5
Slide-PVT-S [37]	-	251G	42.4	63.9	45.0	26.8	45.6	56.9
Pyramid ViG-S [15]	36M	240G	41.8	63.1	44.7	28.5	45.4	53.4
PViHGNN-S [16]	38M	244G	42.2	63.8	45.1	**29.3**	45.9	**55.7**
DVHGNN-S (ours)	38M	242G	**43.3**	**64.3**	**46.3**	28.3	**47.9**	54.6

Backbone	Mask R-CNN 1×							
	Param	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet-50 [17]	44M	260G	38.0	58.6	41.4	34.4	55.1	36.7
PVT-Small [49]	44M	245G	40.4	62.9	43.8	37.8	60.1	40.3
Swin-T [32]	48M	264G	42.2	64.6	46.2	39.1	61.6	42.0
Slide-PVT-S [37]	42M	269G	42.8	65.9	46.7	40.1	63.1	43.1
ConvNeXt-T [33]	48M	262G	44.2	66.6	48.3	40.1	63.3	42.8
Pyramid ViG-S [15]	46M	259G	42.6	65.2	46.0	39.4	62.4	41.6
PViHGNN-S [16]	48M	262G	43.1	66.0	46.5	39.6	63.0	42.3
DVHGNN-S (ours)	49M	261G	**44.8**	**66.8**	**49.0**	**40.2**	**63.5**	**43.1**

Table 2. Results of object detection and instance segmentation on COCO 2017. The input size is 1280 × 800.

Backbone	Method	Params (M)	FLOPs (G)	mIoU (%)
ResNet-50 [17]	S-FPN	29	183	36.7
Swin-T [32]	S-FPN	32	182	41.5
PVT-S [49]	S-FPN	28	225	42.0
Slide-PVT-S [37]	S-FPN	26	188	42.5
DAT-T [53]	S-FPN	32	198	42.6
InceptionNeXt-T [60]	S-FPN	28	-	43.1
DVHGNN-S (ours)	S-FPN	32	181	**43.8**
Swin-T [32]	UperNet	60	945	44.5
FLaTeN-Swin-T [14]	UperNet	60	946	44.8
DAT-T [53]	UperNet	60	957	45.5
Slide-Swin-T [37]	UperNet	60	946	45.7
ConvNeXt-T [33]	UperNet	60	945	46.1
DVHGNN-S (ours)	UperNet	60	945	**46.8**

Table 3. Results of semantic segmentation on ADE20K validation set. The input size is 2048 × 512.

Step	Method	Params (M)	FLOPs (G)	Top-1 acc (%)
0	Feature Dispatching	7.7	2.1	76.1
1	More Heads & Thinner	11.0	1.7	76.1
2	DHConv	11.1	1.8	$77.2_{(+1.1)}$
3	+DHGC	11.1	1.8	$78.0_{(+0.8)}$
4	++ConvFFN	11.2	1.9	$78.4_{(+0.4)}$

Table 4. Ablation experiments of DVHGNN, where "Feature Dipatching", "DHConv" and "DHGC"epresent Feature Dipatching in baseline model, Dynamic Hypergraph Convolution, Dilated Hypergraph Construction. The training epochs is set to 250.

bones. Specifically, under the RetinaNet framework, our DVHGNN-S achieves 43.3% mAP that surpasses ViG by 1.5% and ViHGNN by 1.1%, respectively. Under the Mask R-CNN framework, our DVHGNN-S achieves 44.8% bbox mAP and 40.2% mask mAP that surpasses ViHGNN by 1.7% and 0.6%, respectively.

4.3. Semantic Segmentation on ADE20K

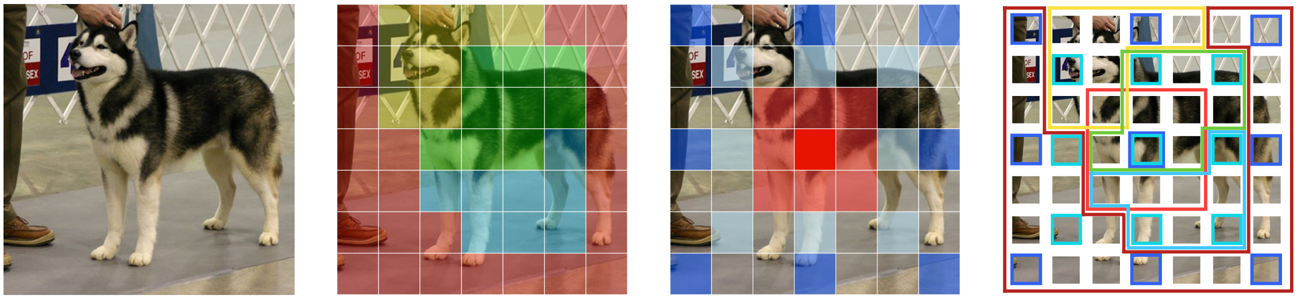
Experimental Settings: We evaluated the semantic segmentation performance of our DVHGNN on ADE20K [69] dataset using two representative frameworks: Upernet [54] and Semantic FPN [25], with our pre-trained DVHGNN-S as the backbone. For Upernet, we followed the Swin Transformer [32] configuration and trained our model for 160K iterations. As for Semantic FPN with 80K iterations, we followed the PVT [49] configuration.

Experimental Results: Table. 3 shows the superior results of our DVHGNN with other popular backbones on the ADE20K validation set. Specifically, using the Semantic FPN [25] framework, our DVHGNN-S achieves 43.8% mIoU, surpassing Swin-T [32] by 2.3 %. Under UperNet [54] framework, our DVHGNN-S achieves 46.8% mIoU, surparssing Swin-T [32] by 2.3 %.

4.4. Ablation Study

Ablation study of modules. In our structured ablation experiments on the ImageNet-1K dataset, as depicted in Table. 4, we trained the DVHGNN-T model for 250 epochs to systematically evaluate the contributions of its modules. In Step 0, we aligned the channel configuration and downsampling stages of the baseline model, ContextCluster-Ti [36], with those of DVHGNN-T, achieving a Top-1 accuracy of 76.1%. In Step 1, we increased the number of heads while making the model thinner and deeper, which reduced the computational cost by 19% GFLOPs. In Step 2, the integration of our Dynamic Hypergraph Convolution, replacing the standard Feature Dispatching mechanism, led to a 0.9% improvement in Top-1 accuracy. Step 3 involved the incorporation of dilated hyperedges via DHGC, further boosting accuracy by 0.8%. Finally, in Step 4, replacing the conventional MLP with the ConvFFN architecture resulted in an additional 0.3% increase in Top-1 accuracy.

Ablation Study of Hyperparameters. To rigorously assess the impact of key hyperparameters on DVHGNN performance, we conducted comprehensive ablation studies on four crucial factors: (1) the number of clustering centroids C, (2) the number of attention heads per stage h, (3) the feature dimension of each head D', and (4) the minimum kernel size (dilation factors). All experiments were conducted using the DVHGNN-T variant, trained on ImageNet-1K. As



(a) Input image.

(b) Hyperedges of Clustering.

(c) Hyperedges of DHGC.

(d) Hypergraph structure.

Figure 5. Visualization of the hypergraph structure of DVHGNN. The hypergraph structure is obtained by an overlay of hyperedges derived from the Clustering method and DHGC.

C	Params (M)	FLOPs (G)	Top-1 (%)	Top-5 (%)
4	11.2	1.9	**79.8**	**95.3**
9	11.3	1.95	79.6	95.0
16	11.5	2.0	79.5	94.9
25	13.0	2.1	79.7	95.1

Table 5. Effects of number of clustering centroids C (300 epochs).

h	Params (M)	FLOPs (G)	Top-1 (%)
[2, 4, 8, 16]	10.2	1.76	74.8
[3, 6, 12, 24]	11.2	1.92	75.8
[4, 8, 16, 32]	12.2	2.08	**76.3**

Table 6. Effects of number of heads h (200 epochs).

D'	Params (M)	FLOPs (G)	Top-1 (%)	Top-5 (%)
16	10.4	1.75	75.3	92.6
24	11.2	1.92	75.8	93.1
32	12.0	2.10	**76.1**	**93.3**

Table 7. Effects of dimension of each head D' (200 epochs).

Minimum Kernel size	Params (M)	FLOPs (G)	Top-1 (%)
3×3	11.2	1.87	75.5
5×5	11.2	1.89	75.6
7×7	11.2	1.92	**75.8**

Table 8. Effects of different Mininum Kernel sizes (200 epochs).

depicted in Table. 5, increasing the number of clustering centroids beyond a certain threshold leads to a marginal decline in Top-1 accuracy, likely due to redundancy and over-clustering, which degrades feature aggregation. Thus, we adopt $C = 4$ as the optimal setting. Conversely, Table. 6 and Table. 7 demonstrate that increasing the number of attention heads h and the feature dimension per head D' consistently enhances performance, underscoring the benefits of richer representation learning. Moreover, as indicated in Table. 8, employing a 7×7 minimum kernel size yields the best results, emphasizing the advantages of larger receptive fields for spatial feature extraction.

4.5. Visualization

To provide a more intuitive evaluation of our proposed DVHGNN model, we visualize the hypergraph construction of one head in the DVHGNN-S model after the last stage. As depicted in Figure. 5, the background areas on the left, right, and bottom are grouped into one hyperedge (represented by red patches), while distinct components of the dog, including its head, body, and limbs, are assigned to separate hyperedges (represented by yellow, green, and blue patches, respectively). This observation highlights that our model not only effectively captures global semantic structures but also comprehends intricate intra-class relationships among object parts, demonstrates the strength of DVHGNN in learning high-level semantic information, enabling accurate and structured object understanding.

5. Conclusion

In this paper, we have proposed a novel vision backbone architecture, termed the DVHGNN, to learn hypergraph-aware vision features. The proposed method is characterized by its dynamic and learnable hypergraph, which can adaptively capture the multi-scale dependencies of an image. Furthermore, a novel dynamic hypergraph convolution is designed to aggregate vertex features into hyperedge features. The extensive qualitative and quantitative experimental results on the benchmark vision datasets demonstrate that the proposed DVHGNN significantly enhances the learning performance of different vision tasks, which achieves remarkable Top-1 accuracy on the ImageNet-1K. In future work, we will further explore the scalability and generalization of the proposed across the other vision tasks.

Acknowledgments The work was supported by the National Science Fund for Distinguished Young Scholars (No.62025603) and National Natural Science Foundation of China (No. 62072386), supported by Yunnan Provincial Major S&T Special Plan Project (No. 202402AD080001), Henan Key R&D Project (No. 231111212000), Henan Center for Outstanding Overseas Scientists (No. GZS2022011), Open Foundation of Henan Key Lab of General Aviation Technology (No. ZHKF-230212), Key Lab of Oracle Information Processing of MOE (No. OIP2024E002). We also thank Digital Culture Lab of Tencent SSV for their invaluable contribution during our research.

References

[1] Devanshu Arya, Das Gupta, Stevan Rudinac, and Marcel Worring. Hypersage: Generalizing inductive representation learning on hypergraphs. *Cornell University - arXiv*, 2021. 3

[2] Guikun Chen, Xia Li, Yi Yang, and Wenguan Wang. Neural clustering based visual representation learning. In *CVPR*, 2024. 6

[3] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[4] Yu Chen, Lingfei Wu, and MohammedJ. Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Neural Information Processing Systems*, 2020. 3

[5] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPRW*, pages 702–703, 2020. 6

[6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017. 1

[7] Zihang Dai, Hanxiao Liu, QuocV. Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *arXiv: Computer Vision and Pattern Recognition*, 2021. 2

[8] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *CVPR*, pages 11963–11975, 2022. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2

[10] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. *Cornell University - arXiv*, 2019. 3

[11] Yue Gao, Meng Wang, Dacheng Tao, Rongrong Ji, and Qionghai Dai. 3-d object retrieval and recognition with hypergraph analysis. *IEEE Transactions on Image Processing*, page 4290–4303, 2012. 2

[12] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, pages 729–734. IEEE, 2005. 2

[13] WilliamL. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Neural Information Processing Systems*, 2017. 2

[14] Dongchen Han, Xuran Pan, Yizeng Han, Shiji Song, and Gao Huang. Flatten transformer: Vision transformer using focused linear attention. In *ICCV*, pages 5961–5971, 2023. 6, 7

[15] Kai Han, Yunhe Wang, Jianyuan Guo, Yehui Tang, and Enhua Wu. Vision gnn: An image is worth graph of nodes. In *NeurIPS*, pages 8291–8303. Curran Associates, Inc., 2022. 2, 3, 6, 7

[16] Yan Han, Peihao Wang, Souvik Kundu, Ying Ding, and Zhangyang Wang. Vision hgnn: An image is more than a graph of nodes. In *ICCV*, pages 19878–19888, 2023. 2, 3, 6, 7

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1, 2, 6, 7

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017. 6

[19] 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. 1, 2

[20] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, pages 646–661. Springer, 2016. 6

[21] Jing Huang and Jie Yang. Unignn: a unified framework for graph and hypergraph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021. 5

[22] Yuchi Huang, Qingshan Liu, and Dimitris Metaxas.]video object segmentation by hypergraph cut. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 2

[23] Yuchi Huang, Qingshan Liu, Shaoting Zhang, and Dimitris N. Metaxas. Image retrieval via probabilistic hypergraph ranking. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010. 2

[24] Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. Dynamic hypergraph neural networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019. 3, 5

[25] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, pages 6399–6408, 2019. 7

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pages 1097–1105, 2012. 1, 2

[27] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 2

[28] James C Liang, Yiming Cui, Qifan Wang, Tong Geng, Wenguan Wang, and Dongfang Liu. Clusterformer: Clustering as a universal visual learner. *NeurIPS*, 2023. 1, 6

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 6

[30] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 6

[31] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model. *arXiv preprint arXiv:2401.10166*, 2024. 6

[32] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021. 1, 2, 4, 6, 7

[33] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, pages 11976–11986, 2022. 2, 7

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[35] Dongsheng Luo, Wei Cheng, Wenchao Yu, Bo Zong, Jingchao Ni, Haifeng Chen, and Xiang Zhang. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021. 3

[36] Xu Ma, Yuqian Zhou, Huan Wang, Can Qin, Bin Sun, Chang Liu, and Yun Fu. Image as set of points. *ICLR*, 2023. 5, 6, 7

[37] Xuran Pan, Tianzhu Ye, Zhuofan Xia, Shiji Song, and Gao Huang. Slide-transformer: Hierarchical vision transformer with local self-attention. In *CVPR*, pages 2082–2091, 2023. 1, 7

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 32, 2019. 6

[39] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115:211–252, 2015. 6

[40] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1): 61–80, 2008. 2

[41] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, page 93, 2017. 2

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 2

[43] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 1, 2

[44] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016. 6

[45] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114. PMLR, 2019. 1

[46] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pages 10347–10357. PMLR, 2021. 1, 2

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017. 1, 2, 5

[48] Nikil Wale and George Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. In *Sixth International Conference on Data Mining (ICDM'06)*, 2006. 2

[49] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, pages 568–578, 2021. 1, 2, 7

[50] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved baselines with pyramid vision transformer. *CVM*, 8(3):415–424, 2022. 6

[51] Ross Wightman et al. Pytorch image models, 2019. 6

[52] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22–31, 2021. 2

[53] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *CVPR*, pages 4794–4803, 2022. 1, 7

[54] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. *Unified Perceptual Parsing for Scene Understanding*, page 432–448. 2018. 7

[55] Hang Xu, Chenhan Jiang, Xiaodan Liang, and Zhenguo Li. Spatial-aware graph relation network for large-scale object detection. In *CVPR*, pages 9298–9307, 2019. 2

[56] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks. 2018. 5

[57] Yufei Xu, Qiming Zhang, Jing Zhang, and Dacheng Tao. Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Neural Information Processing Systems*, 2021. 2

[58] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *ECCV*, pages 684–699, 2018. 2

[59] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *CVPR*, pages 10819–10829, 2022. 6

[60] Weihao Yu, Pan Zhou, Shuicheng Yan, and Xinchao Wang. Inceptionnext: When inception meets convnext. *2024 IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 6, 7

[61] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pages 6023–6032, 2019. 6

[62] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 6

[63] Jiying Zhang, Yuzhao Chen, Xi Xiao, Runiu Lu, and Shu-Tao Xia. Learnable hypergraph laplacian for hypergraph learning. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022. 3

[64] Zizhao Zhang, Haojie Lin, Yue Gao, and A. Dynamic hypergraph structure learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018. 3

[65] Zizhao Zhang, Yifan Feng, Shihui Ying, and Yue Gao. Deep hypergraph structure learning. 2022. 3

[66] Wei Zhao, Shulong Tan, Ziyu Guan, Boxuan Zhang, Maoguo Gong, Zhengwen Cao, and Quan Wang. Learning to map social network users by unified manifold alignment on hypergraph. *IEEE Transactions on Neural Networks and Learning Systems*, page 5834–5846, 2018. 2

[67] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. *International Conference on Machine Learning*, 2020. 3

[68] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, pages 13001–13008, 2020. 6

[69] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7

[70] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Forty-first International Conference on Machine Learning*. 6