Hypergraph Vision Transformers: Images are More than Nodes, More than Edges

Joshua Fixelle
University of Virginia
jf9fk@virginia.edu

Abstract

*Recent advancements in computer vision have highlighted the scalability of Vision Transformers (ViTs) across various tasks, yet challenges remain in balancing adaptability, computational efficiency, and the ability to model higher-order relationships. Vision Graph Neural Networks (ViGs) offer an alternative by leveraging graph-based methodologies but are hindered by the computational bottlenecks of clustering algorithms used for edge generation. To address these issues, we propose the **H**yper**g**raph **V**ision **T**ransformer (**HgVT**), which incorporates a hierarchical bipartite hypergraph structure into the vision transformer framework to capture higher-order semantic relationships while maintaining computational efficiency. HgVT leverages population and diversity regularization for dynamic hypergraph construction without clustering, and expert edge pooling to enhance semantic extraction and facilitate graph-based image retrieval. Empirical results demonstrate that HgVT achieves strong performance on image classification and retrieval, positioning it as an efficient framework for semantic-based vision tasks.*

1. Introduction

Computer vision has recently transitioned from the historically dominant Convolutional Neural Networks (CNNs) [20, 28, 30] to the increasingly prominent Vision Transformers (ViTs), which have quickly embedded themselves as the new de facto standard [12, 38]. This shift reflects the broader success of transformers in natural language processing [11, 54, 56] and is driven by the remarkable scalability of ViTs across various tasks such as image classification [53, 61], semantic segmentation [26, 63], and image retrieval [2, 29]. While hybrid models like hierarchical attention and CNN-ViT methods [18, 19, 33] aim to balance computational load and flexibility, ViTs still struggle with focusing on salient features rather than comprehensive image understanding [2, 9, 12, 40]. This highlights the need for methods that improve both efficiency and semantic accuracy.

Within the spectrum of novel architectures, Vision Graph Neural Networks (ViGs) [16] and Vision Hypergraph Neural Networks (ViHGNNs) [17] leverage graph-based topologies to advance image processing. Unlike CNNs, which harness locality and translation-invariance through densely connected pixel grids and repeated convolutions, both ViTs and ViGs represent images as sets of patches. In ViTs, each patch acts as a vertex within a maximally connected graph, creating semantically weak connections through self-attention. ViGs enhance this by using clustering algorithms to detect edge groupings and applying graph convolutions to these clusters, forming meaningful patch relationships. ViHGNNs extend these capabilities by employing hyperedges that capture complex, higher-order relationships, enriching understanding of the images. These methodologies are depicted in Figure 1.

While graph-based models like ViG and ViHGNN have introduced notable advancements in visual perception, two critical observations emerge regarding these architectures:

1. In existing vision GNN models [16, 17, 35, 36], edge features are primarily used for basic vertex-to-vertex communication and are not integrated across successive layers: a strategy that could enhance cumulative learning and improve classification accuracy.
2. The computational complexities associated with clustering algorithms used for edge generation, such as KNN in ViG and Fuzzy C-Means in ViHGNN, pose significant computational bottlenecks. Approaches like MobileViG [35] and GreedyViG [36] attempt to mitigate these challenges with static graph structures and adding dynamic masking, but do so by trading adaptability for efficiency, failing to achieve a well-balanced solution.

In response to limitations in existing graph-based models, we propose the Hypergraph Vision Transformer (HgVT), which advances the hypergraph concept with a bipartite representation where hyperedge features and image patches (vertices) are continuously processed. Unlike traditional models that use graph convolutions, HgVT employs structured multi-head attention for efficient vertex-hyperedge message passing and incorporates a dynamic querying mechanism that constructs graph structures in $\mathcal{O}(|V| \cdot E)$ time complexity, where $E < |V|$. This graph structure is then utilized in attention masking to balance structural adaptability with computational efficiency. Furthermore, HgVT integrates

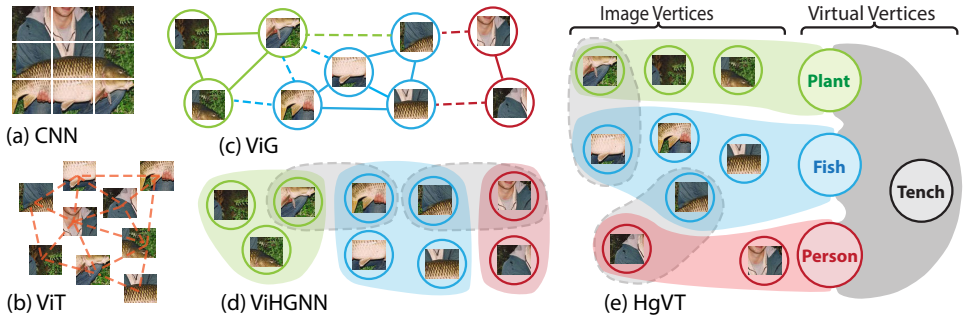


Figure 1. Comparison of Graph Structures for different methods. Showing (a) CNNs, (b) Vision Transformers, (c) ViG with a KNN clustered GNN, (d) ViHGNN with clustered hyperedges, and (e) our proposed HgVT method. Strong group edges shown with solid lines; weak edges with dashed lines. Hyperedges shown with shaded regions; less dominant hyperedges with gray dashed regions.

virtual elements into vertices and hyperedges, forming intermediate links that create a hierarchical semantic structure, as illustrated in Fig. 1e. Attention masking is then used to restrict message passing, and virtual hyperedge features are leveraged for classification. Our contributions are as follows:

- We propose the Hypergraph Vision Transformer (HgVT), which integrates a hierarchical bipartite hypergraph structure within a vision transformer framework. Our isotropic HgVT-Ti model achieves a Top-1 accuracy of 76.2% on the ImageNet-1k classification task, surpassing the prior state-of-the-art by 1.9%, demonstrating the efficacy of hypergraph-based learning within vision transformers.
- We introduce population and diversity regularization strategies that enable dynamic hypergraph structure construction in HgVT, allowing the model to self-sparsify connections without relying on traditional clustering techniques.
- We implement expert edge pooling, a pooling approach that selects edges based on learned confidence scores, facilitating efficient representation pruning and graph extraction. This approach demonstrates strong semantic clustering behavior across macro-classes and achieves competitive image retrieval performance compared to other feature extractors, while maintaining a more compact model size.

2. Related Work

Vision Transformers. Vision Transformers (ViTs) proposed by [12] and refined by [2, 38, 53] use self-attention to process image patches as sequences, scaling to complex datasets and tasks. Recent ViTs have reintroduced spatial hierarchies by leveraging local attention [19, 33], integrating sparse global summaries [68], and employing biomimetic modeling to focus on key regions within images [49]. However, current models tend to focus on the most salient objects and patch-level similarities, ignoring global structure. HgVT addresses this by introducing bipartite hypergraphs to model higher-order relationships for improved semantic understanding.

Graph-Based Vision Models and Clustering. Graph Neural Networks (GNNs), initially conceptualized by [46], have been applied to vision tasks through Vision Graph Neural Networks (ViGs) [16], which exhibit improved accuracy over ViTs on common vision tasks. ViGs use graph convolutions to model image patch relationships on a graph structure, typically constructed by iterative clustering algorithms such as KNN and Fuzzy C-Means, which introduce computational overhead. Recent methods avoid clustering inefficiencies with static graph structures [35, 36], sacrificing adaptability. HgVT instead introduces a dynamic graph construction method, relying on cosine similarity from learned features to enable efficient, non-iterative, adaptive clustering.

Hypergraph-Based Methods. While previously used in many computer vision tasks [15, 23, 24], hypergraphs have recently been incorporated into vision GNNs [17, 50], improving their ability to model complex multi-way relationships. However, these methods treat hypergraphs as an intermediate tool rather than producing a hypergraph to represent underlying images, preventing their use in downstream tasks. HgVT instead iteratively refines a hypergraph through subsequent network layers to produce structured representations.

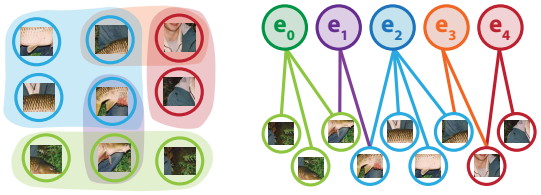
3. Hierarchical Hypergraphs

Graphs and Basic Notations. Graphs are powerful mathematical tools for representing structured information, applicable across diverse disciplines. A graph $\mathcal{G}$ is defined as a pair $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is a set of vertices, and $\mathcal{E} = \{e_{ij} | (v_i, v_j)\}$ is a set of directed edges. For undirected graphs, the edges are symmetric, such that $e_{ij} = e_{ji}$. Each edge e_{ij} connects a pair of vertices v_i and v_j, where $v_i, v_j \in \mathcal{V}$. The adjacency matrix $\hat{\mathbf{A}}$ is a binary matrix $\{0,1\}^{|V|\times|V|}$, representing the presence (1) or absence (0) of an edge between each pair of vertices. Similarly, an edge weight matrix can be defined as $\mathbf{A} \in \mathbb{R}^{|V|\times|V|}$ to quantify the strength or capacity of these connections.

Graph Convolution Networks (GCNs). Building on this foundation, GCNs utilize vertex feature matrices $\mathbf{X} \in \mathbb{R}^{|V|\times d}$ to encode vertex properties. Their core mechanism, message passing, updates vertex features through a convolution

with a learned projection matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ and a non-linear activation, guided by the adjacency matrix $\hat{\mathbf{A}}$, which specifies neighboring vertices. Adjacency features $\mathbf{X}_{\text{adj}} \in \mathbb{R}^{|V| \times d_a}$, typically set as $\mathbf{X}_{\text{adj}} = \mathbf{X}$, enable dynamic updates to $\hat{\mathbf{A}}$ and the edge weight matrix $\mathbf{A}$, refining the graph structure through learned interactions. However, GCNs are unable to model multi-vertex relationships as edges in $\mathcal{E}$ are pairwise.

3.1. Hypergraphs and Bipartite Representations



(a) Hypergraph (b) Bipartite Form

Figure 2. Comparison of (a) hypergraph and (b) equivalent bipartite representation from Fig. 1d, showing five hyperedges (e_0 – e_4).

To overcome the pairwise limitation inherent in traditional graphs, hypergraphs offer a robust solution by extending the concept of edges to hyperedges, which connect multiple vertices simultaneously. In a hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, hyperedges $e_j \in \mathcal{E}$ each connect a subset of vertices, defined as $e_j = \{v_i \mid v_i \in \mathcal{V} \text{ and } i \in I_j\}$, where I_j is the set of indices for vertices that are included in hyperedge e_j. The set I_j directly corresponds to the nonzero entries of the j-th column of the incidence matrix $\mathbf{H} \in \{0,1\}^{|V| \times |E|}$, where $\mathbf{H}_{ij} = 1$ if vertex v_i is included in the hyperedge e_j. This structure captures complex relationships, making hypergraphs valuable for modeling networked systems and group interactions.

Hypergraphs can alternatively be described using a bipartite representation, where the vertex set $\mathcal{V}$ and hyperedge set $\mathcal{E}$ form distinct groups linked by the incidence matrix $\mathbf{H}$ (refer to Fig. 2). This representation results in a new graph $\mathcal{G}_B = (\mathcal{V}, \mathcal{E}, \mathcal{E}_B)$, where $\mathcal{V}$ represents the original vertices of the hypergraph, and the elements in $\mathcal{E}$ correspond to hyperedges. The edges in $\mathcal{E}_B$, denoted as $\epsilon_{ve} = (\nu_v, \nu_e)$ exist if $\mathbf{H}_{ve} = 1$, with $\nu_v \in \mathcal{V}$ and $\nu_e \in \mathcal{E}$, linking $\mathcal{V}$ and $\mathcal{E}$.

In the bipartite graph $\mathcal{G}_B$, the corresponding adjacency matrix can be simplified as $\hat{\mathbf{A}} = \mathbf{H}$ for $\mathcal{E} \to \mathcal{V}$ interactions, and $\hat{\mathbf{A}} = \mathbf{H}^T$ for $\mathcal{V} \to \mathcal{E}$. Drawing on principles similar to those in ViHGNN [17], the edge weight matrix $\mathbf{A}$ can be interpreted as fuzzy membership weights, enabling graded interactions across GNN layers. Complementing this setup, the feature matrices are split into $\mathbf{X}^{(V)} \in \mathbb{R}^{|V| \times d_v}$ and $\mathbf{X}^{(E)} \in \mathbb{R}^{|E| \times d_e}$, along with their correspond adjacency feature matrices $\mathbf{X}^{(V)}_{\text{adj}}$ and $\mathbf{X}^{(E)}_{\text{adj}}$, mirroring traditional GNNs.

3.2. Imposing Hierarchical Structure in Images

To enhance the capability of hypergraphs in image analysis, we draw inspiration from the register tokens introduced in [8], which act to summarize information that otherwise manifests as noise in areas of low visual significance. Similarly, this work integrates virtual vertices ($v\mathcal{V}$), alongside typical image patch vertices ($i\mathcal{V}$), and introduces virtual hyperedges ($v\mathcal{E}$), alongside primary hyperedges ($p\mathcal{E}$), providing levels of semantic feature aggregation and relational abstraction. Notably, these virtual elements, illustrated in Fig. 1e, do not correspond to specific image patches; instead, they are learned embeddings used for semantic summarization and capturing high-level abstract information.

Our proposed hypergraph, constructed from image I as $\mathcal{G}_B(I)$, integrates primary and virtual sets, forming $\mathcal{V} = i\mathcal{V} \cup v\mathcal{V}$ and $\mathcal{E} = p\mathcal{E} \cup v\mathcal{E}$, with statically masked communication pathways to enforce a hierarchical structure. Primary hyperedges ($p\mathcal{E}$) interact with all vertices to support unrestricted semantic aggregation, whereas virtual hyperedges ($v\mathcal{E}$), designated for class predictions, connect solely with virtual vertices ($v\mathcal{V}$). These restrictions separate visual and abstract information, thereby producing a graph structure suitable for use in downstream applications.

4. A Hypergraph Vision Transformer

The Hypergraph Vision Transformer (HgVT) adapts the architecture of standard Vision Transformers by incorporating bipartite hypergraph features for enhanced image analysis capabilities. Like Vision Transformers, HgVT begins with a patch embedding layer, followed by an isotropic stack of $L\times$ HgVT blocks, culminating in feature pooling and a classifier head. The bipartite hypergraph is represented by four principal feature matrices – $\mathbf{X}^{(V)}$, $\mathbf{X}^{(V)}_{\text{adj}}$, $\mathbf{X}^{(E)}$, and $\mathbf{X}^{(E)}_{\text{adj}}$ – which are updated iteratively and in an interleaved fashion within each block. Each block constructs a new adjacency matrix $\mathbf{A}$ from $\mathbf{X}^{(V)}_{\text{adj}}$ and $\mathbf{X}^{(E)}_{\text{adj}}$, enabling flexible adjustments to the hypergraph structure. As illustrated in Fig. 3, this modular process allows for the continuous integration and processing of these matrices within each HgVT block.

4.1. Hyperedges as Communication Pools

Each HgVT block processes both vertex and edge information, refining them from the previous block based on a newly constructed graph structure. Initially, adjacency mask computation (detailed in the next section) determines the connectivity for the subsequent processing steps within each block, dynamically adjusting to the updated feature matrices from the previous block. Three attention layers – vertex self-attention, edge aggregate attention, and edge distribution attention – operate sequentially to enhance feature integration and facilitate effective message passing along the hyperedges formed in the adjacency computation step. Finally, separate feed-forward networks process vertex and edge features independently, ensuring specialized treatment for the two distinct sets within the bipartite hypergraph, preserving the unique properties of each set. Operational details of these components are further described in Appendix A.

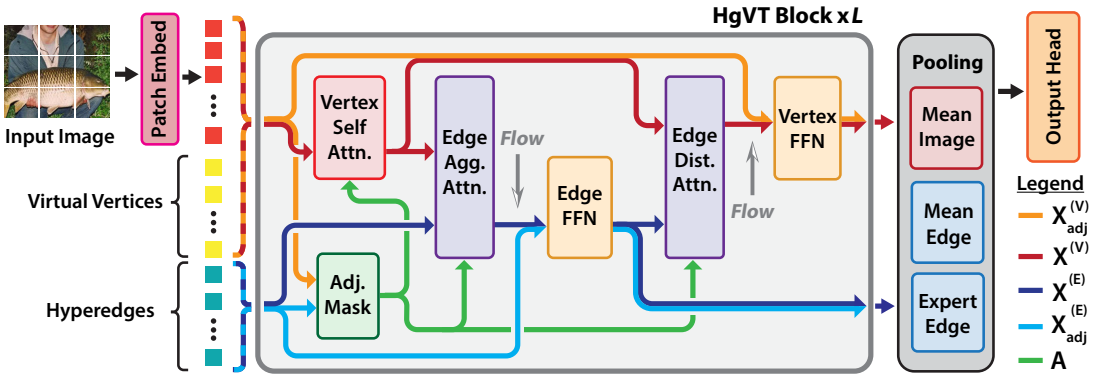
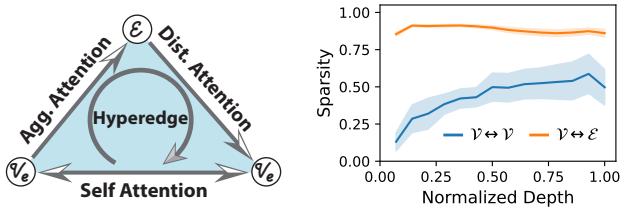


Figure 3. HgVT Architecture Diagram, composed of stacked HgVT blocks with adjacency matrix $\mathbf{A}$, vertex features $\mathbf{X}^{(V)}$, and hyperedge features $\mathbf{X}^{(E)}$. Edge attention flow is shown with gray arrows; input norms and residual adds are omitted for clarity.

Hypergraph Feature Processing. Within each HgVT block, two distinct point-wise feed-forward networks (FFNs) independently process vertex and hyperedge features, aligning with the bipartite structure of the hypergraph. Each FFN integrates both the element features and their corresponding adjacency features through a fully connected layer, improving the model's ability to synthesize relationships. Processing both feature types within the same FFN layer allows adjacency information to be handled directly, bypassing the need for graph-based message passing and improving computational efficiency. Moreover, parameter overhead can be reduced by optionally tying edge and vertex FFN weights.



(a) Communication Pool. (b) Attention Sparsity.

Figure 4. (a) Hyperedge Communication Pool Flow with edges $\mathcal{E}$ and member vertices $\mathcal{V}_e$; (b) Attention Sparsity (Mean and std) for HgVT-S on the ImageNet-1k Validation set.

Hyperedges as Communication Pools. Hypergraph GNNs typically employ a gather→scatter mechanism for processing vertex-hyperedge interactions, whereas HgVT reconceptualizes hyperedges as communication pools that facilitate information flow among vertices and their associated hyperedges. Specifically, vertex self-attention manages vertex-to-vertex ($\mathcal{V} \rightarrow \mathcal{V}$) interactions within hyperedges, edge aggregate attention orchestrates the flow from vertices to hyperedges ($\mathcal{V} \rightarrow \mathcal{E}$), and edge distribution attention handles the reverse, from hyperedges back to vertices ($\mathcal{E} \rightarrow \mathcal{V}$). By segmenting the attention operations, HgVT efficiently approximates an all-to-all feature transfer within hyperedges, as illustrated in Fig. 4a, significantly reducing the quadratic complexity associated with full attention mechanisms.

Sparse and Fuzzy Attention. Building upon the dynamic communication pools concept, HgVT employs both sparse and fuzzy attention mechanisms to further optimize computational efficiency. Vertex self-attention is applied selectively to pairs of vertices connected by common hyperedges, as defined by the adjacency matrix $\hat{\mathbf{A}}$, resulting in a sparse attention pattern. As sparsity increases with network depth – demonstrated in Fig. 4b – computational load decreases, while still maintaining compatibility with dense attention during training. Conversely, the edge aggregate and distribution attention mechanisms utilize cross-attention between the vertex and edge feature matrices, $\mathbf{X}^{(V)}$ and $\mathbf{X}^{(E)}$, modulated by the soft adjacency matrix $\mathbf{A}$. This modulation, akin to Fuzzy C-Means in ViHGNNs [17], adjusts attention logits based on soft memberships to the individual hyperedges, dynamically adapting to the hypergraph structure and providing a mechanism for gradient flow into the adjacency matrix generation. Furthermore, by thresholding the soft adjacency matrix during inference, the edge attention mechanisms can be converted into a sparse cross-attention mechanism, thereby reducing computational overhead.

4.2. Dynamic Adjacency Formation

HgVT dynamically establishes its hypergraph structure to adapt to the varying semantic and spatial structures of different image inputs. It employs cosine similarity, akin to query-key interactions in attention mechanisms, to evaluate the alignment between vertex and hyperedge adjacency features. This approach allows hyperedges to "query" vertices for relevant features, providing a scale-invariant assessment that emphasizes the directionality of embedding vectors. The cosine similarity is subsequently transformed into adjacency membership using a sharpened sigmoid function:

$$\mathrm{A} = \sigma\left(\alpha \cdot \tilde{\mathbf{X}}^{(V)}_{\mathrm{adj}} \left[\tilde{\mathbf{X}}^{(E)}_{\mathrm{adj}}\right]^{T}\right), \quad \tilde{\mathbf{X}}^{(*)}_{\mathrm{adj}} = \frac{\mathbf{X}^{(*)}_{\mathrm{adj}}}{||\mathbf{X}^{(*)}_{\mathrm{adj}}||_2 + \epsilon} \quad (1)$$

Here, σ denotes the sigmoid function and $\alpha = 4$ is a sharpening factor, which pushes values away from zero to estab-

lish binary-like membership values in matrix $\mathbf{A}$. This soft adjacency matrix is further thresholded to create the hard adjacency matrix $\hat{\mathbf{A}} = [\mathbf{A} > 0.5]$, which provides binary memberships to facilitate sparse attention masking.

4.3. Architecture Scaling

Table 1. Scaling variants of our HgVT architecture. All models are trained at 224x224 resolution, except lite variant (HgVT-Lt), trained at 160x160. Showing count for vertices ($i\mathcal{V}$, $v\mathcal{V}$), hyperedges ($p\mathcal{E}$, $v\mathcal{E}$), dim for adj. (d_a) and features (d_f), depth (L), and heads (h).

Model	$\lvert i\mathcal{V}\rvert$	$\lvert v\mathcal{V}\rvert$	$\lvert p\mathcal{E}\rvert$	$\lvert v\mathcal{E}\rvert$	$d_f + d_a$	L	h	Params	FLOPS
HgVT-Lt	100	12	32	6	$128+64$	12	4	6.8M	0.92B
HgVT-Mi	196	16	50	8	$128+64$	9	4	5.8M	1.39B
HgVT-Ti	196	16	50	8	$128+64$	12	4	7.7M	1.80B
HgVT-S	196	16	50	8	$224+96$	14	7	23M	5.48B

HgVT adopts a hybrid scaling strategy inspired by DeiT [53] and ViG [16], with Tab. 1 detailing the transformer scaling hyperparameters and the vertex and edge type allocations, including fixed capacities for non-image vertices ($i\mathcal{V}$) as proposed by ViHGNN [17]. Our scaling approach aims to balance parameter count and FLOPs with the two main GNN baselines, ViG and ViHGNN. We also introduce a Mini variant (HgVT-Mi) to match DeiT-Ti's computational profile, which we achieve by repeating the final block twice for an effective depth of 9; see Appendix H for details. Finally, we include a Ti-Lite variant (HgVT-Lt) for more efficient ablation studies under a constrained training budget.

5. Enforcing Semantic Structure

Feature matrices for virtual vertices and hyperedges, lacking direct input-based initialization, risk converging to homogeneous solutions and collapsed representations. Additionally, the dynamic adjacency calculation alone fails to naturally promote semantic grouping, in contrast to clustering-based approaches commonly used in vision GNNs. To address these issues, we introduce diversity regularization, which enforces orthogonal embeddings, and population regularization, which encourages a structured, sparse hypergraph. For enhanced semantic differentiation of virtual hyperedge features, we further incorporate an expert-based pooling strategy as a more robust alternative to mean pooling.

5.1. Diversity-Driven Feature Differentiation

To prevent homogenization of learned feature matrices and to encourage distinct, semantically rich embeddings, we implement a diversity-driven regularization approach. This method, designed to maintain maximum orthogonality among the embeddings of virtual vertices and hyperedges, penalizes the absolute value of the cosine similarity between different feature vectors, aiming for values close to zero. By using normalized embeddings and masking off diagonal elements to preserve self-similarity, the approach prevents the model from converging to homogeneous solutions or driving individual vectors towards zero magnitude. We then individually penalize $v\mathcal{V}$, $\mathcal{E}$, and their adjacency features.

$$\mathrm{D_L}\left(\mathbf{X}\right) = \frac{1}{2}\sum_{ij}\left(1-\delta_{ij}\right)\cdot\left|\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T\right|_{ij},\ \tilde{\mathbf{X}} = \frac{\mathbf{X}}{||\mathbf{X}||_2+\epsilon} \quad (2)$$

$$\mathcal{L}_{\mathrm{DIV}} = \sum_{x}\mathrm{D_L}\left(x\right),\ x\in\{\mathbf{X}^{(:vV)},\mathbf{X}^{(:vV)}_{\mathrm{adj}},\mathbf{X}^{(E)},\mathbf{X}^{(E)}_{\mathrm{adj}}\} \quad (3)$$

Where $(:vV)$ represents the subset of $\mathcal{V}$ containing only the virtual nodes, δ_{ij} is the Kronecker delta function, ensuring that self-similarity is not penalized, and $|\cdot|_{ij}$ denotes the element-wise absolute value, applied to calculate the penalty for non-orthogonal relationships between embeddings.

5.2. Population Regularization: Learned Sparsity

Unlike clustering methods such as KNN or Fuzzy C-Means, which impose fixed cluster sizes, our model's dynamic adjacency calculation enables flexible and self-adjusting hyperedge populations. To prevent the associated risks of overly sparse or densely connected hypergraphs, we introduce population regularization. This method applies penalties based on the computed soft membership density of each hyperedge derived from the soft adjacency matrix $\mathbf{A}$, ensuring each maintains a vertex population within appropriate bounds to avoid overgeneralization and preserve hypergraph integrity.

$$P_j = 2\cdot\sum_i \max(A_{ij}-0.5,0) \quad (4)$$

$$\mathcal{L}_{\mathrm{POP}} = \sum_j \max(P_j-\beta,0)+\max(\gamma-P_j,0) \quad (5)$$

Here, P_j represents a soft density estimate of vertex connections for the j^{th} hyperedge, only considering non-zero entries of $\hat{\mathbf{A}}$. The hyperparameters β and γ set the upper and lower density limits, ensuring that hyperedges maintain an optimal balance of connections. Penalties are applied if P_j either exceeds β or falls below γ, maintaining the desired sparsity and ensuring the structural efficacy of the hypergraph.

5.3. Expert Pooling for Semantic Specialization

To effectively combine features from multiple virtual hyperedges for classification, our approach utilizes a method akin to expert-choice, where each virtual hyperedge acts as an "expert" generating a confidence score. Unlike mean pooling, which risks collapsing distinct features into an average that may dilute individual contributions, this strategy encourages virtual hyperedges to develop unique, semantically meaningful representations. The normalized confidence scores, $P(e)$, determine the relevance of each hyperedge e's contribution to the classification task.

$$P(e) = \mathrm{softmax}\left(\mathbf{X}_e^{(:vE)}\mathbf{W}_e + b_e\right) \quad (6)$$

Here, $(:vE)$ denotes the subset of (E) containing only the virtual hyperedges, and the softmax is computed across the

expert gate set $\{e\}$ after projection by $\mathbf{W} \in \mathbb{R}^{d \times |:vE|}$. During training, $P(e)$ guides the weighted averaging of hyperedge features before subsequent class prediction. We further apply a density loss function [3, 14] to prevent underutilization of any single virtual hyperedge, complemented by a cross-entropy term with label smoothing to increase expert confidence. During inference, top-k routing combines the most relevant hyperedges based on their confidence scores.

6. Empirical Evaluation and Performance

This section evaluates the Hypergraph Vision Transformer using two model configurations from Tab. 1: HgVT-Ti-Lite for targeted ablation studies and scaled variants for benchmarking against comparable classifiers. We apply standard augmentations from DeiT [53] using the Timm library [58], including RandAugment [7], Mixup [64], Cutmix [60], Random Erasing [66], and Repeated Augment [21].

Datasets. For classification in computer vision, we follow standard practices and use the ImageNet-1k dataset [10] at a resolution of 224x224 pixels for scaled model evaluations. For ablation studies, we employ ImageNet-100 [51], a 100-class subset of ImageNet-1k with images scaled to 160x160 pixels. This selection provides a computationally manageable dataset while maintaining sufficient class variation and larger image sizes compared to datasets like CIFAR-100 (32x32 pixels)[27]. Nevertheless, we find CIFAR-100 useful for assessing the effects of regularization on the hypergraph structure, as detailed in Appendix H.

Training Hyperparameters. Consistent with DeiT, we use the AdamW optimizer with a weight decay of 0.05. Training is conducted on the ImageNet-1k dataset with a batch size of 1024 for 300 epochs following DeiT. For ablations, we train on ImageNet-100 with a batch size of 512 for with a shorter duration of 200 epochs as proposed by [31]. Learning rates follow a cosine-annealing schedule peaking at 1e-3 for both datasets following scaling from DeiT. Furthermore, we omit the use of Exponential Moving Average (EMA) due to its minimal performance improvement (0.1% in DeiT) relative to its overhead per training step. All models were trained using PyTorch on NVIDIA A6000 GPUs; see Appendix I.

Evaluation Metrics. Following standard protocols, we measure the Top-1 class prediction accuracy to assess overall performance. Additionally, we take advantage of the learned graph structure (extracted from the last layer) on each image, and measure: Hyperedge Entropy (HE), Intra-Cluster Similarity (ICS), Inter-Cluster Distance (ICD), and Silhouette Score (SIL) [45]; further details on structure in Appendix C.

6.1. Evaluation on ImageNet

Tab. 2 presents the ImageNet-1k Top-1 accuracy of HgVT, benchmarked against isotropic models. We limit evaluation to isotropic models due to the complexity of down-sampling virtual tokens, lacking direct spatial correspondence, needed for pyramidal structures. As a result, we also exclude other pyramidal models from our comparison as they typically outperform isotropic models within the same architecture family [16, 17], making the comparison less meaningful.

Table 2. ImageNet-1k results for HgVT and other isotropic networks. ❄ CNN, ♦Transformer, ★GNN, ■HGNN, and ▲HgVT.

Model	Params	FLOPs	ImNet Top-1	ReaL Top-1	V2 Top-1
❄ ResMLP-S12 conv3x3 [52]	16.7M	3.2B	77.0	84.0	65.5
❄ ConvMixer-768/32 [55]	21.1M	20.9B	80.2	–	–
❄ ConvMixer-1536/20 [55]	51.6M	51.1B	81.4	–	–
♦DINOv1-S [2]	21.7M	4.6B	77.0	–	–
♦ViT-B/16 [12]	86.4M	55.5B	77.9	83.6	–
♦DeiT-Ti [53]	5.7M	1.3B	72.2	80.1	60.4
♦DeiT-S [53]	22.1M	4.6B	79.8	85.7	68.5
♦DeiT-B [53]	86.4M	17.6B	81.8	86.7	71.5
★ViG-Ti [16]	7.1M	1.3B	73.9	–	–
★ViG-S [16]	22.7M	4.5B	80.4	–	–
★ViG-B [16]	86.8M	17.7B	82.3	–	–
■ViHGNN-Ti [17]	8.2M	1.8B	74.3	–	–
■ViHGNN-S [17]	23.2M	5.6B	**81.5**	–	–
■ViHGNN-B [17]	88.1M	19.4B	82.9	–	–
▲HgVT-Mi (ours)	5.8M	1.4B	**74.4**	**81.8**	**62.7**
▲HgVT-Ti (ours)	7.7M	1.8B	**76.2**	**83.2**	**64.3**
▲HgVT-S (ours)	22.9M	5.5B	**81.2**	**86.7**	**70.1**

HgVT-Ti demonstrates a clear advantage, surpassing ViHGNN-Ti by 1.9% in accuracy, with the smaller HgVT-Mi model matching ViHGNN-Ti with only 70% of the parameters. At the S-scale, we reduce HgVT's depth relative to ViHGNN to accommodate hyperedge attention and adjacency feature overhead, aiming for comparable parameter counts; as a result, HgVT-S only matches ViHGNN-S. Despite this limitation, HgVT-S matches DeiT-B's accuracy on ImageNet ReaL [1] and achieves competitive performance on ImageNet V2 [43], at nearly a quarter of DeiT-B's size.

Overall, HgVT achieves strong performance on classification, demonstrating the benefits of integrating hypergraph structures within a vision transformer framework. Additionally, computational evaluation in Appendix B shows a consistent $\geq 1.4\times$ speedup over ViG, with adjacency computation exhibiting a $10\times$ improvement over ViG's KNN-based method. These improvements further highlight HgVT's potential for greater efficiency over prior Vision GNNs.

6.2. Ablation Studies

We conducted a series of ablations on the ImageNet-100 dataset using the HgVT-Lt model, reporting Top-1 classification accuracy alongside mean hyperedge entropy and silhouette scores to assess the quality of the hypergraph's structure. Notably, we observe a weak anti-correlation between the graph quality metrics and Top-1 accuracy (see Appendix H), indicating opposing objectives. Overall, the ablations are grouped into three categories: regularization, architecture, and pooling methods, with results in Tab. 3.

Table 3. ImageNet-100 ablations with HgVT-Lt. Indicating used (✓) or not used (✗), and pooling methods: Average (***A***), Image (***I***), Expert (***E***), Expert+Image (***EI***), and ***EI*** dropping ***I*** (***DI***).

Ablation on ↓	CLS Dropout	Stoch. Decay	Diversity	Population	Tied FFN	$\mathbf{X}_{\text{adj}} = \mathbf{X}$	d_f Mult.	Pooling Op	Edge Entropy	Silhouette	Top-1 Acc.	Params (M)
none: HgVT-Lt	✗	✓	✓	✓	✓	✓	***1***	***EI***	3.32	0.780	84.36	6.75
Regularization	✓	✓	✓	✓	✓	✓	1	*E*	3.13	0.751	82.23	6.62
	✗	✗	✓	✓	✓	✓	1	*E*	3.12	0.745	81.89	6.62
	✗	✓	✗	✓	✓	✓	1	*E*	1.99	0.723	80.79	6.62
	✗	✓	✓	✗	✓	✓	1	*E*	3.89	0.639	81.79	6.62
	✗	✓	✗	✗	✓	✓	1	*E*	3.58	0.610	81.99	6.62
Architecture	✗	✓	✓	✓	✗	✓	1	*E*	3.09	0.741	82.89	9.86
	✗	✓	✓	✓	✗	✗	1	*E*	2.27	0.808	78.62	5.84
	✗	✓	✓	✓	✓	✗	1	*E*	2.12	0.780	76.95	4.40
	✗	✓	✓	✓	✓	✗	***1.5***	*E*	2.05	0.770	77.46	9.62
Pooling	✗	✓	✓	✓	✓	✓	1	***A***	3.07	0.747	82.06	6.62
	✗	✓	✓	✓	✓	✓	1	***I***	2.93	0.760	84.08	6.62
	✗	✓	✓	✓	✓	✓	1	***E***	3.13	0.749	82.52	6.62
	✗	✓	✓	✓	✓	✓	1	***DI***	3.32	0.780	80.94	6.75

As shown in Tab. 3, the regularization ablations demonstrate that *stochastic path dropout decay* [22] improves both Top-1 accuracy and silhouette scores, consistent with ViG and ViHGNN [16, 17]. Omitting *Class dropout* also boosts accuracy, aligning with DeiT [53]. Futhermore, our proposed *diversity* and *population* regularization are essential for preserving graph structure; removing diversity leads to partial representation collapse, while removing population regularization results in near-zero sparsity, effectively turning HgVT into a ViT with increased network complexity.

In the *architecture* ablations, untying the FFN improves accuracy but significantly increases parameter count, making it an unfavorable tradeoff. Tying adjacency and embedding features ($\mathbf{X}_{\text{adj}} = \mathbf{X}$) reduces parameters and FLOPs but degrades performance, and while untying the FFN or increasing feature dimensionality partially mitigates this, the parameter increase remains suboptimal. This indicates that adjacency and embedding features ($\mathbf{X}_{\text{adj}}^{(*)}$ and $\mathbf{X}^{(*)}$) are similar, yet require dedicated feature spaces to avoid crowding.

For *pooling methods*, *expert edge* pooling outperforms *average edge* pooling in accuracy, while *image* pooling achieves the highest accuracy at the cost of degraded graph structure. Combining image and expert pooling recovers lost structure and improves accuracy, with each input focusing on different semantic levels (see Appendix D). Additionally, dropping the pooled image embedding prior to the classifier head maintains moderate performance, indicating that both paths meaningfully contribute to the final prediction.

Impact of Vertex Self-Attention and Patch Embedding. We evaluated the impact of *patch embedding* methods and *vertex self-attention*, comparing a convolutional stem (Conv2D-BN-GELU layers [16, 17]) and a simpler patch projection (pixel-shuffled patches with affine projection [2, 12]), across various pooling strategies (average, image, and expert), as shown in Tab. 4. The patch projection consistently underperforms the convolutional stem, likely due to the model's small size limiting its effectiveness. Omitting vertex self-attention leads to further degradation, especially without the convolutional stem, suggesting it is crucial for effectively separating features in low-dimensional space. PCA shows that 71/128 channels are needed to explain 95% of the variance for the convolutional stem, compared to 19/128 for the patch projection, indicating the richer representation captured by the convolutional stem. Notably, image pooling shows the least degradation, likely due to its more direct gradient flow compared to the edge pooling methods.

Table 4. Top-1 accuracy of HgVT-Lt on ImageNet-100 with (✓) or without (✗) vertex self-attention. Contrasting pooling operations and patch embedding versions: Conv. Stem or Patch Projection.

Pooling Op. →	Average		Image		Expert	
Vertex SA →	✗	✓	✗	✓	✗	✓
Conv. Stem	78.02	82.06	82.05	84.08	78.87	82.52
Patch Project	64.30	72.76	70.76	76.17	62.62	71.43

6.3. Pooling Methods and Graph Structure

Table 5. Impact of graph structure on pooling operation for HgVT-Lt on ImageNet-100. Measuring graph metrics for image vertices ($i\mathcal{V}$) and all vertices ($\mathcal{V}$), along with features from DINOv2.

Pooling Op. →	Image			Expert			Img. & Expert		
Feature Model ↓	HE	ICS	ICD	HE	ICS	ICD	HE	ICS	ICD
HgVT-Lt ($i\mathcal{V}$)	3.03	0.50	0.31	3.24	0.43	0.37	3.23	0.42	0.36
HgVT-Lt ($\mathcal{V}$)	3.20	0.25	0.72	3.39	0.36	0.45	3.39	0.28	0.58
DINO2-S ($i\mathcal{V}$)	3.04	0.84	0.08	3.25	0.85	0.05	3.25	0.83	0.06
DINO2-G ($i\mathcal{V}$)	3.04	0.69	0.15	3.25	0.68	0.12	3.25	0.68	0.13

To evaluate the impact of pooling methods on graph structure, we measured HE, ICS, and ICD using the ImageNet-100 validation set with three strategies: image pooling, expert pooling, and a combined image + expert pooling approach. Metrics used either the image vertex subset ($i\mathcal{V}$) or the full vertex set ($\mathcal{V}$), with features derived from DINOv2 (S and G) [38] and HgVT-Lt using HgVT-Lt's adjacency matrix ($\mathbf{A}$). Results in Tab. 5 show that while all methods achieve similar graph quality using $i\mathcal{V}$, image pooling slightly improves similarity. However, including all vertices ($\mathcal{V}$) consistently increases ICD and entropy while reducing ICS, indicating decreased graph coherence. This effect is more severe with image pooling methods, suggesting that virtual vertices ($v\mathcal{V}$) act as noisy elements, rather than summarization points.

Comparing DINOv2 models, all pooling methods align more closely with DINOv2-G, where achieving a balance between ICS and ICD is preferable to maximizing either individually. This trend, along with consistent HE, suggests a focus on higher-level detail, irrespective of the smaller HgVT model size or pooling method. Image pooling shows slightly stronger alignment with both DINO models, indicating that

both high- and low-level semantics are encoded within a single feature space, unlike methods that can use edge channels for high-level concepts. Notably, all expert pooling methods exhibit an emergent macro-class prediction behavior, where each virtual edge ($v\mathcal{E}$) consistently captures broader taxonomic groups (e.g., dogs, birds). Further representation and macro-class analysis are provided in Appendices D and J.

6.4. Performance on Image Retrieval

To evaluate the capability of HgVT in capturing semantic structures, we perform image retrieval experiments comparing four methods: pooling similarity (PS), volumetric similarity (VS), adaptive pooling similarity (APS), and adaptive volumetric similarity (AVS). PS ranks the pooled embeddings by cosine similarity (vector search), while the other methods enhance retrieval by leveraging graph structure. Volumetric similarity calculates ellipsoid overlap using an approximate Mahalanobis distance, with pruned hyperedges defining the distribution spread around the pooled embedding (centroid). Adaptive methods further refine retrieval via a graph similarity measure on the pruned hyperedges, re-ranking from a shortlist of $R = 100$. Computational efficiency in adaptive retrieval is ensured through centroid hash binning and limiting comparisons to the top-$C = 4$ most significant query hyperedges, resulting in a complexity of $\mathcal{O}(RC)$. Notably, we prune to 12 hyperedges and use 10 centroid bins. Additional details provided in Appendix G.

Table 6. Image Retrieval on ImageNet-1k with proposed search methods: PS, APS, VS, and AVS. Reporting Top-1 accuracy, mAP@10 (%), and 1-NN-hit@10 (%) for pooling methods.

mAP@10 (%)						
Model	Top-1	Pooling	PS	APS	VS	AVS
MRL-128 [29]	70.52	Image	64.94	65.20	–	–
MRL-256 [29]	70.62	Image	65.04	65.20	–	–
HgVT-Ti (ours)	76.18	Expert	70.56	69.59	70.53	69.40
HgVT-Ti (ours)	76.18	Im&Ex	73.23	69.59	73.08	69.49
1-NN-hit@10 (%)						
HgVT-Ti (ours)	76.18	Expert	21.22	19.10	21.25	19.56
HgVT-Ti (ours)	76.18	Im&Ex	25.13	19.10	25.22	19.17

ImageNet Retrieval. We evaluate retrieval performance on the ImageNet-1K dataset to assess HgVT's ability to capture semantic relationships and compare four retrieval methods: PS, VS, APS, and AVS. The primary metric is mAP@10, with MRL [29] serving as the baseline due to its adaptive re-ranking approach. We also report the 1-NN-CLIP-L hit-rate@10, which measures how often the top-1 result ranked by CLIP-L [42] appears in the top-10 retrieved results, offering additional insight into semantic alignment. Results in Tab. 6 show that HgVT-Ti surpasses MRL by over 8% in retrieval performance, despite being significantly smaller than MRL (ResNet-50) and using a comparably compact embedding size ($d = 2\times128$). Among our methods, PS and VS achieve similar results, while APS and AVS underperform, likely due to their focus on exact-feature similarity and ambiguous-class features, which limits alignment with the high-level semantics required by ImageNet's diverse dataset.

Table 7. Image Retrieval on revisited Oxford and Paris; reporting mAP (%). Showing training set and method used. *mAP@100.

		Dataset →	$\mathcal{R}$Oxford		$\mathcal{R}$Paris	
Model	Train Set	Method	M	H	M	H
ALEX+GeM [44]	ImNet-1k	PS	33.8	10.4	52.7	26.0
RN101+R-MAC [44]	ImNet-1k	PS	49.8	18.5	74.0	52.1
DINOv1-S/16 [2]	ImNet-1k	PS	41.8	13.7	63.1	34.4
DINOv1-S/16 [2]	GLDv2	PS	51.5	24.3	75.3	51.6
HgVT-Ti (ours)	ImNet-1k	VS	26.8	10.5	55.4	28.7
		VS*	25.8	9.0	65.6	28.0
		AVS*	27.0	6.8	64.1	26.7
HgVT-S (ours)	ImNet-1k	VS	28.0	12.1	56.7	31.1
		VS*	26.3	10.2	65.0	28.4
		AVS*	27.4	10.3	65.0	27.1

Oxford and Paris Retrieval. To evaluate image retrieval performance beyond simple class retrieval, we use the Oxford and Paris Revisited datasets [39, 41], which provide three splits of increasing difficulty (Easy, Medium, and Hard) for query/database pairs. We report Mean Average Precision (mAP) for the Medium (M) and Hard (H) splits, using mAP@100 for AVS based on short-list ranking, while full mAP and mAP@100 are provided for VS as a baseline comparison. Results, shown in Tab. 7, indicate that HgVT achieves competitive performance with similarly sized feature extractors, though performance on $\mathcal{R}$Oxford-M lags. This shortfall may stem from subtle landmark differences better captured by multi-scale Conv-Nets and self-supervised learning, compared to the more salient focus driven by HgVT's classifier training. However, AVS outperforms VS on $\mathcal{R}$Oxford-M, demonstrating its ability to uncover finer feature similarities within the hypergraph structure.

7. Conclusion and Future Directions

In this work, we introduced the Hypergraph Vision Transformer (HgVT), which combines hypergraph structures with vision transformers. Our methods, including diversity and population regularization, along with expert edge pooling, enhance semantic representation and efficiency by enabling dynamic hypergraph construction. Although challenging to tune, these methods avoid reliance on traditional clustering techniques, reflecting a tradeoff between complexity and efficiency. Empirical results show that HgVT achieves strong performance on both image classification and retrieval, positioning HgVT as a competitive framework for semantic vision tasks. Future work will focus on exploring the scalability of hypergraph structures and integrating self-supervised learning to further improve adaptability and better decouple saliency from semantic vision graph generation.

References

[1] Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? *arXiv preprint arXiv:2006.07159*, abs/2006.07159, 2020. 6

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jegou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9630–9640, 2021. 1, 2, 6, 7, 8, 4, 5

[3] Tianlong Chen, Zhenyu Zhang, Ajay Kumar Jaiswal, Shiwei Liu, and Zhangyang Wang. Sparse moe as the new dropout: Scaling dense and self-slimmable transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. 6

[4] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 15

[5] Norman Cliff. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological Bulletin*, 114:494–509, 1993. 8

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 14

[7] Ekin Dogus Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3008–3017, 2019. 6

[8] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. 3

[9] Dmitry Demidov, M.H. Sharif, Aliakbar Abdurahimov, Hisham Cholakkal, and Fahad Shahbaz Khan. Salient mask-guided vision transformer for fine-grained classification. In *VISIGRAPP*, 2023. 1

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 6

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019. 1

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 2, 6, 7, 4, 5

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 14

[14] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23:120:1–120:39, 2022. 6

[15] 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*, 21(9): 4290–4303, 2012. 2

[16] Kai Han, Yunhe Wang, Jianyuan Guo, Yehui Tang, and Enhua Wu. Vision GNN: An image is worth graph of nodes. In *Advances in Neural Information Processing Systems*, 2022. 1, 2, 5, 6, 7, 4

[17] Yan Han, Peihao Wang, Souvik Kundu, Ying Ding, and Zhangyang Wang. Vision hgnn: An image is more than a graph of nodes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19878–19888, 2023. 1, 2, 3, 4, 5, 6, 7, 27

[18] Yunusa Haruna, Shiyin Qin, Abdulrahman Hamman Adama Chukkol, Abdulganiyu Abdu Yusuf, Isah Bello, and Adamu Lawan. Exploring the synergies of hybrid cnns and vits architectures for computer vision: A survey. *arXiv preprint arXiv:2402.02941*, abs/2402.02941, 2024. 1

[19] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2

[20] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015. 1, 15

[21] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8126–8135, 2020. 6

[22] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Weinberger. Deep networks with stochastic depth, 2016. 7

[23] Yuchi Huang, Qingshan Liu, and Dimitris Metaxas.]video object segmentation by hypergraph cut. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1738–1745, 2009. 2

[24] 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*, pages 3376–3383, 2010. 2

[25] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. 15

[26] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3992–4003, 2023. 1

[27] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 6

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012. 1

[29] Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham M. Kakade, Prateek Jain, and Ali Farhadi. Matryoshka representation learning. In *Neural Information Processing Systems*, 2022. 1, 8

[30] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[31] Sangjun Lee, Inwoo Hwang, Gi-Cheon Kang, and Byoung-Tak Zhang. Improving robustness to texture bias via shape-focused augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 4323–4331, 2022. 6

[32] Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xformers: A modular and hackable transformer modelling library. https://github.com/facebookresearch/xformers, 2022. 28

[33] 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 *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021. 1, 2, 15

[34] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. 9

[35] Mustafa Munir, William Avery, and Radu Marculescu. Mobilevig: Graph-based sparse attention for mobile vision applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2211–2219, 2023. 1, 2

[36] Mustafa Munir, William Avery, Md Mostafijur Rahman, and Radu Marculescu. Greedyvig: Dynamic axial graph construction for efficient vision gnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6118–6127, 2024. 1, 2

[37] NVIDIA. Apex: A pytorch extension. https://github.com/NVIDIA/apex, 2020. 28

[38] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 1, 2, 7, 6, 8, 14, 15

[39] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 8

[40] Joan Puigcerver, Carlos Riquelme Ruiz, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts. In *The Twelfth International Conference on Learning Representations*, 2024. 1

[41] Filip Radenovic, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5706–5715, 2018. 8

[42] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. 8, 6

[43] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, 2019. 6

[44] Jérôme Revaud, Jon Almazán, Rafael Sampaio de Rezende, and César Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5106–5115, 2019. 8

[45] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. 6, 7

[46] 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, 2009. 2

[47] Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, abs/2002.05202, 2020. 4

[48] Dai Shi. Transnext: Robust foveal visual perception for vision transformers. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17773–17783, 2023. 14, 15

[49] Dai Shi. Transnext: Robust foveal visual perception for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17773–17783, 2024. 2

[50] Sakhinana Sagar Srinivas, Rajat Kumar Sarkar, Sreeja Gangasani, and Venkataramana Runkana. Vision HgNN: An electron-micrograph is worth hypergraph of hypernodes. *arXiv preprint arXiv:2408.11351*, abs/2408.11351, 2024. 2

[51] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020. 6

[52] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, and Hervé Jégou. Resmlp: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404*, abs/2105.03404, 2021. 6

[53] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers amp; distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 1, 2, 5, 6, 7, 4

[54] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, abs/2302.13971, 2023. 1, 2

[55] Asher Trockman and J. Zico Kolter. Patches are all you need? *Trans. Mach. Learn. Res.*, 2023, 2023. 6

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 1

[57] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. 4

[58] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 6, 28

[59] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *European Conference on Computer Vision*. Springer, 2018. 15

[60] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Young Joon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031, 2019. 6

[61] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 1204–1213. IEEE, 2022. 1

[62] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 2

[63] Bowen Zhang, Zhi Tian, Quan Tang, Xiangxiang Chu, Xiaolin Wei, Chunhua Shen, and Yifan liu. Segvit: Semantic segmentation with plain vision transformers. In *Advances in Neural Information Processing Systems*, 2022. 1

[64] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. 6

[65] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2016. 15

[66] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *AAAI*, 34: 13001–13008, 2020. 6

[67] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127:302 – 321, 2016. 14

[68] Lei Zhu, Xinjiang Wang, Zhanghan Ke, Wayne Zhang, and Rynson Lau. Biformer: Vision transformer with bi-level routing attention. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2