TECHNICAL REPORT 2023

Hierarchical Graph Convolutional Networks for Image Classification

(Submitted – conference – BRACIS)

03

TECHNICAL REPORT

2023

Hierarchical Graph Convolutional Networks for Image Classification

CITATION DETAILS:
João Pedro Batistelli, Silvio Jamil F. Guimarães, Zenilton Kleber Gonçalves do Patrocínio Jr. (2023). Hierarchical Graph Convolutional Networks for Image Classification. Technical Report ImScience/PUC Minas #03/2023.

03

Technical report – ImScience 03/2023

Hierarchical Graph Convolutional Networks for Image Classification

João Pedro Oliveira Batisteli · Silvio Jamil F. Guimarães · Zenilton K. Gonçalves do Patrocínio Jr.

June 8, 2023

Abstract Graph-based image representation is a promising research direction that can capture the structure and semantics of images. However, existing methods for converting images to graphs often fail to preserve the hierarchical information of the image elements and produce sub-optimal or poor regions. To address these limitations, we propose a novel approach that uses a hierarchical image segmentation technique to generate graphs at multiple segmentation scales, reflecting the hierarchical relationships between image elements. We also propose and train a **H**ierarchical **G**raph **C**onvolutional Network for **I**mage **C**lassification (HGCIC) model that leverages the hierarchical information with three different adjacency setups on the CIFAR-10 database. Experimental results show that the proposed approach can achieve competitive or superior performance compared to other state-of-the-art methods while using smaller graphs.

Keywords Deep Learning · Graph Neural Networks · Image Classification.

1 Introduction

Graph-based image representation is an emerging research area that leverages the spatial relationships between image elements to model image content more effectively [16]. Graph-based approaches can enhance the understanding of image semantics and context by incorporating domain-specific knowledge into the learning process. Moreover, these approaches can provide multi-scale representations of the same image [17, 22, 23], capturing local and global information about its structure. These advantages make graph-based image representation appealing for various image analysis tasks.

Silvio Jamil F. Guimarães
ImScience/PUC-Minas – Belo Horizonte 31980-110, Brazil
E-mail: sjamil@pucminas.br

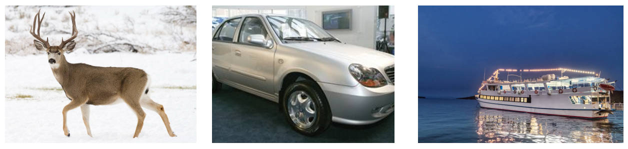


Fig. 1 Examples of images classified as airplanes, according to [2].

Applying machine learning algorithms to graph data presents unique challenges due to its irregularity, variable sizes, and diverse neighbor relationships [24]. Graph Neural Networks (GNNs) have emerged to address these challenges, adapting neural network architectures to process graph-structured data effectively. GNNs capture the intricate relationships and dependencies between vertices in a graph, leveraging information from neighboring vertices and edges to encode local and global structural information. This approach enables accurate predictions and improved generalization, making GNNs ideal for tackling various graph-based machine-learning problems.

One of the main applications of GNNs is graph-based image analysis, which requires representing images as graphs. However, this is a challenging task. One common possibility in literature is to apply image segmentation methods for partitioning the image into regions and representing each region by a vertex in a graph [2, 9, 17, 19, 22, 23]. Despite the simplicity of this approach, it has two important drawbacks: (i) the dependency on the quality and quantity of the regions produced by the segmentation algorithm; and (ii) the hierarchical relationship between image elements are not captured. Here we argue that this property is essential for effective image reasoning, which a graph representation must capture. It is worth mentioning that hierarchical information could be seen as a multi-scale representation. Figure 1 illustrates failures in image classification when the hierarchical structure is disregarded (all cases were classified as airplanes, using [2] representation).

To overcome the limitations of existing methods, we propose a **novel approach that leverages hierarchical segmentation techniques to generate graph vertices for image classification from graph representation**. Hierarchical segmentation is a well-established technique in computer vision and image processing [6,7,13], enabling the identification of objects and regions within an image based on their visual characteristics. Additionally, our method allows incorporating hierarchical relationships between image elements into the resulting graph, facilitating image analysis tasks. More specifically, the idea of hierarchical segmentation is similar to generating a set of image segmentations at different levels of detail with respect to the principles of multi-scale image analysis [12]. By adopting this approach, we argue that we can effectively capture the rich structural information in the image and encode it in the resulting graph. Thus, our proposal for image classification uses a hierarchical segmentation method in order to capture hierarchical information for representing image data features. However, in some cases, while working on raw images, noise may produce poor results in image segmentation, for that, in this work, instead of using raw images, we generate superpixels from them to produce homogeneous and concise regions in conjunction with good object delineation. From these superpixels, region adjacency graphs are computed for representing superpixel images.

To evaluate the effectiveness of our proposed representation, we conducted experiments using images from the CIFAR-10 dataset, leveraging our hierarchical segmentation technique from the superpixel images to generate graph representations that were subsequently used to train the Hierarchical Graph Convolutional Network for Image Classification (HGCIC) model. Test experiments demonstrated promising results, even though the graphs used to train our model were smaller than those utilized in other works. The proposed approach outperformed the results obtained by some of the state-of-the-art methods, highlighting its effectiveness.

We can briefly describe the two major contributions of this work to graph-based image analysis: (i) the proposition of a novel graph representation method that leverages hierarchical image segmentation to capture hierarchical representations of the underlying image structure; and (ii) the introduction of a novel graph convolutional neural network architecture that can extract and use the essential information from our new graph representation.

This work is organized as follows. Section 2 presents some related works. Section 3 describes the most important concepts needed for understanding the proposed method. Section 4 presents a hierarchical graph convolutional network for classifying images based on superpixel graphs. Section 5 describes the experiments and some comparative analysis of the proposed approach to the state-of-the-art methods. And, finally, in Section 6, we have drawn some conclusions and present some further work.

2 Related Works

Superpixel image segmentation could be seen as a fundamental task in image processing that divides an image into homogeneous regions. This process can reduce the complexity of an image and provide a more efficient representation for further analysis. By treating these regions or segments as vertices, we can transform the image into a graph structure, which can facilitate the use of graph-based algorithms.

A common strategy in literature for representing images as graphs is based on considering vertices as superpixels [2, 9, 11, 17, 20, 22]. Superpixels are groups of pixels into perceptually meaningful regions based on similarity criteria like color or location. This approach leverages the flexibility of graph neural networks, which can handle irregular graphs with different shapes and sizes. Other methods include splitting the images into fixed patches viewed as vertices [14] or interpreting each pixel as a vertex [8, 20]. Some works also explore multiscale graph (from image) representations to achieve better graph classification performance.

Multiscale graph-image representations can capture different levels of details and features from the image. For instance, in [22], the authors proposed a novel superpixel algorithm that produces segments with a wide size distribution, allowing a more flexible representation of an image, as it can capture fine and coarse details. The authors in [23] model hyperspectral images as multiple graphs with different neighborhood scales and propose a dynamic graph convolution operation that updates the similarity measures among vertices by fusing feature embeddings. To the best of our knowledge, the work in [17] is the only one that has used a multiscale graph segmentation, in which superpixels are obtained at several scales and different types of relations between vertices are explored to improve the model expressiveness.

The edges of the graph are also crucial for constructing the graph representation, as they enable the message-passing mechanism of GNNs. The message-passing mechanism updates the feature representations of vertices by exchanging information with neighboring vertices through edges. One natural way to build adjacency is using a region adjacency graph (RAG), in which edges are created between spatial neighbors [2, 22]. Another strategy is to construct k-nearest adjacency based on the spatial and/or feature distances of the vertices, as used in [9, 14]. Both methods have shown promising results, and their choice mainly depends on the problem requirements and the properties of the data.

3 Theoretical Background

3.1 Graph Neural Networks

Graph neural networks (GNNs) can be divided into two types: spatial or spectral. Spatial GNNs work directly on the graph structure and compute vertex (and maybe edge) representations using information from their neighbors. Spectral GNNs, on the other hand, function on the graph spectral domain. They utilize the eigenvectors of the graph Laplacian matrix as the foundation for representing the graph data. This work focuses solely on spatial GNNs.

The input of each GNN layer are the vertex feature vectors $\{h_u \in \mathbb{R}^d \mid u \in \mathcal{V}\}$, the set of edges $\mathcal{E}$, and optionally edge feature vectors (maybe seen as weights) $\{w_{uv} \in \mathbb{R}^d \mid (u,v) \in \mathcal{E}\}$. The result of each layer is a new vertex representation $\{h_u^{'} \in \mathbb{R}^{d^{'}} \mid u \in \mathcal{V}\}$, in which the same parametric function is applied to each vertex given its neighbors $\mathcal{N}_u = \{v \in \mathcal{V} \mid (u,v) \in \mathcal{E})\}$ and on the edges incident to it, generically given by:

$$h'_u = f_\theta\left(h_u, aggregate(h_v, w'_{uv} \mid v \in \mathcal{N}_u)\right) \quad (1)$$

in which *aggregate* is a permutation invariant function (like max, min, sum), f_θ is the parametric function, and w'_{uv} is the updated edge feature defined by:

$$w'_{uv} = g_\theta(h_u, h_v, w_{uv}) \quad (2)$$

in which g_θ is a distinct parametric function. Each vertex update step is also called the message passing step since vertices send information to their neighbors.

3.2 Residual Gated Graph Convolutional Network

According to [4], GatedGCN is a fusion of the vanilla GCN and edge gating mechanism. In [9], the authors suggested modifications to the GatedGCN architecture by introducing residual connections and batch normalization [15]. The vertex update is given by:

$$h'_u = h_u + \mathrm{ReLU}(\mathrm{BN}(U^\ell h_u + \sum_{v \in N_v} \alpha_{uv} \odot V^\ell h_v)) \quad (3)$$

in which U^ℓ, V^ℓ are linear transformations, $\odot$ denotes Hadamard product, ReLU stands for Rectified Linear Unit, BN represents batch normalization, and α_{uv} are the edge gates defined by:

$$\alpha_{uv} = \frac{\sigma(w_{uv}^{'})}{\sum_{v' \in N_u} \sigma(w_{uv'}^{'}) + \varepsilon} \quad (4)$$

$$w_{uv}^{'} = w_{uv} + \mathrm{ReLU}(\mathrm{BN}(A^\ell h_u^\ell + B^\ell h_v^\ell + C^\ell w_{uv})) \quad (5)$$

in which A^ℓ, B^ℓ, C^ℓ are linear transformations, σ is the sigmoid function, and ε is a small-fixed constant for numerical stability. The edge gate in Equation 4 works as a soft attention mechanism [9], allowing the model to learn the importance of different vertices in a neighborhood.

3.3 Hierarchical Segmentation

Hierarchical image segmentation is a set of image segmentations at different detail levels [13]. The segmentations with lower levels of detail can be created by merging regions from segmentations at higher levels of detail.

Hierarchical approaches must obey the principles of multiscale image analysis. These principles ensure that the segmentation is consistent across different levels of detail. The causality principle defines that a contour presented at a scale k_1 should be present at any scale $k_2 < k_1$. The location principle defines that contours should be stable because they neither move nor deform from one scale to another [12].

Hierarchical image segmentation organizes image segments into a tree structure where each vertex represents a different level of detail or abstraction. The highest level of the tree represents the entire image, while lower levels correspond to smaller and more specific sub-regions or subsegments. This structure provides a way to represent the image at different levels of resolution, allowing for a better understanding of the image's contents.

4 Hierarchical Graph Convolutional Networks by using Hierarchy of Superpixels

Given a finite set V, a *partition* of V is a set $\mathbf{P}$ of nonempty disjoint subsets of V whose union is V. Any element of $\mathbf{P}$, denoted by $\mathbf{R}$, is called a *region* of $\mathbf{P}$. Given two partitions $\mathbf{P}$ and $\mathbf{P'}$ of V, $\mathbf{P'}$ is said to be a (total) refinement of $\mathbf{P}$, denoted by $\mathbf{P'} \preceq \mathbf{P}$, if any region of $\mathbf{P'}$ is included in a region of $\mathbf{P}$. Let $\mathcal{H} = (\mathbf{P}_1, \ldots, \mathbf{P}_\ell)$ be a set of ℓ partitions on V. $\mathcal{H}$ is a hierarchy if $\mathbf{P}_{i-1} \preceq \mathbf{P}_i$, for any $i \in \{2, \ldots, \ell\}$.

4.1 Graph Construction

Let $G = (V, E)$ be a RAG computed from the superpixels in which the set V represents the superpixels and the set E the adjacency relation between the superpixels. Let $\mathcal{H} = (\mathbf{P}_1, \ldots, \mathbf{P}_\ell)$ be a hierarchy computed from the graph G. Let $\mathcal{R}_j$ be the set of regions in the partition $\mathbf{P}_j$ of the hierarchy $\mathcal{H}$. Let $\mathcal{R}$ be set containing all regions belonging to all partition $\mathbf{P}_j \in \mathcal{H}$.

Figure 2 illustrates our proposal for computing the hierarchy from the original image. In the following, we describe how to compute three different graphs from a given hierarchy. These graphs will be used as input in the learning step of our method.

Hierarchy-based graph This graph, deoted by $G_h = (V_h, E_h)$, is a graph computed from the hierarchical structure $\mathcal{H}$ in which the set of vertices V_h is equal to the $\mathcal{R}$. The set of edges E_h is defined by $E_h = \{(r_i, r_j), (r_j, r_i) \mid r_i, r_j \in \mathcal{R},\ r_i \neq r_j,$ in which r_j is the smallest region that contains $r_i\}$.

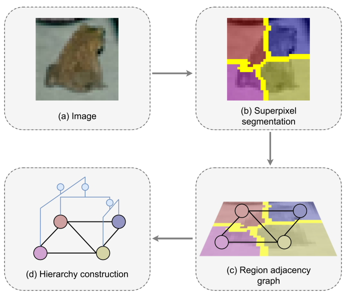

(a) Image
(b) Superpixel segmentation
(c) Region adjacency graph
(d) Hierarchy construction

Fig. 2 Pipeline for computing a hierarchy from the original image.

***k*NN-based graph** This graph, denoted by $G_k = (V_k, E_k)$, is a graph computed from the k-nearest neighborhood of regions in the hierarchical structure $\mathcal{H}$ in the feature space, in which the set of vertices V_k is equal to the $\mathcal{R}$. Let $W(V) = \{w(v),\ \forall v \in V\}$ be the set of feature vectors related to the vertex set V. The set of edges E_k is defined by $E_k = \{(r_i, r_j) \mid w(r_j)$ is one of the k-nn of $w(r_i)\}$.

Complete-based graph This graph, denoted by $G_c = (V_c, E_c)$, is a graph computed from hierarchical structure $\mathcal{H}$, in which the set of vertices V_c is equal to the $\mathcal{R}$. Let $W(V) = \{w(v),\ \forall v \in V\}$ be the set of feature vectors related to the vertex set V. The set of edges E_c is defined by $E_c = \{(r_i, r_j) \mid r_i, r_j \in \mathcal{R},\ r_i \neq\ r_j\}$.

It is important to mention that each region is represented by the following set of features: color (color channels mean and 2-bin color histogram), texture (contrast, dissimilarity, homogeneity, energy, correlation, and angular second moment), region (orientation, bounding box area, solidity, area, eccentricity, convex area, perimeter, mean intensity, Euler number and Hu moments), X and Y mean position and the vertex altitude in the segmentation tree. The extracted features were then used to create a vertice feature vector $h_u \in \mathbb{R}^{104\times 1}$ for each $u \in V$.

4.2 Architecture

Figure 3 shows the proposed GCN architecture for the image classification task. To embed the input edge and vertice features, two linear layers were applied to produce D-dimensional embeddings. The dimension of the edge and vertice embeddings remained the same across all layers. Inspired by [5], this work adopted $\mathcal{M} + 1$ convolutions, in which $\mathcal{M}$ is chosen at inference time. All $\mathcal{M}$ convolutional layers share the same weights, which improved the results and made the proposed architecture parameter-efficient [5].

An adaptive architecture that adjusts its depth can capture important features and patterns that lead to better accuracy and performance. However, balancing model capacity and complexity is crucial to avoid sub-optimal results. If the model is too shallow, it may not be able to capture

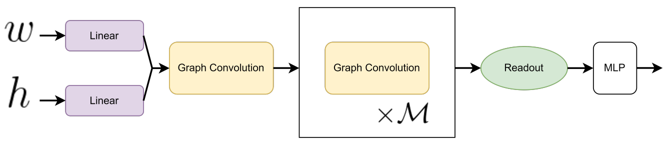


Fig. 3 Model Architecture. The input edges and vertices features are h and w, respectively.

complex patterns, while a model that is too deep may suffer from overfitting or be computationally expensive. After the graph convolution layers, we employ a *readout layer* that generates a fixed-size vector representation from the graph features. The output of the readout layer is then fed into a multi-layer perceptron (MLP) that learns to make class predictions based on the graph features.

By combining the adaptive depth graph convolution layers, readout layer, and MLP, the proposed model can effectively extract and learn hierarchical representations of the input graphs, leading to accurate and robust classification results.

5 Experimental Results

To analyze our proposal for image segmentation, we have applied our strategy to a well-known database. We have considered the three different graphs computed from the hierarchy. Also, we have trained all the models for 1,000 epochs with an initial learning rate of 10^{-3}, which is reduced by half if the validation accuracy does not improve after ten epochs until reaching a stopping learning rate of 10^{-5}, Cross Entropy loss, batch size of 64 and Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$. We saved the weights on the epoch with the best validation accuracy.

We evaluated the proposed model in the CIFAR-10 database [18], comprising 60,000 32×32 color images across ten classes, with 6,000 images each. The database is split into 45,000 training images, 5,000 validation images, and 10,000 test images. It is important to observe that we have followed the procedure described in [9] and randomly sampled 5,000 images from the training set for validation. The same splits were used for all experiments.

5.1 Implementation details

5.1.1 Graph construction

We adopt SLIC method [1] as the superpixel segmentation method since it is simple, fast, and memory efficient, and to make a fair comparison since almost all work in our comparative analysis use it. The target number of superpixels is typically 20 but may vary for each image. We have used the *watershed by area* [6] as the hierarchical segmentation

Table 1 HGSIC Architecture Details

	Input Size	Output Size	# of Parameters	Activation Function
Edge Embedding	$(\lvert\mathcal{E}\rvert$ x 1)	$(\lvert\mathcal{E}\rvert$ x 70)	140	-
Vertex Embedding	$(\lvert\mathcal{V}\rvert$ x 104)	$(\lvert\mathcal{V}\rvert$ x 70)	7,350	-
GatedGCN	$(\lvert\mathcal{E}\rvert$ x 70)	$(\lvert\mathcal{E}\rvert$ x 70)	25,270	ReLU
	$(\lvert\mathcal{V}\rvert$ x 70)	$(\lvert\mathcal{V}\rvert$ x 70)		
GatedGCN	$(\lvert\mathcal{E}\rvert$ x 70)	$(\lvert\mathcal{E}\rvert$ x 70)	25,270	ReLU
	$(\lvert\mathcal{V}\rvert$ x 70)	$(\lvert\mathcal{V}\rvert$ x 70)		
Linear	140	256	36,096	ReLU
Layer Norm.	256	256	512	-
Linear	256	10	2,570	-

Table 2 Accuracy of the proposed model and state-of-art methods in the CIFAR-10 dataset. * means the target number of nodes since the authors do not report the average value, and the cells with – refer to data that have not been reported.

Model	# params	# nodes	# vertices	Accuracy
Spatial GNN				
HGCIC$_k$ (ours)	97,208	47.13	377.04	0.6043
HGCIC$_h$ (ours)	97,208	47.13	92.26	**0.631**
HGCIC$_c$ (ours)	97,208	47.13	2,175.11	0.6186
GAT [2]	55,364	75*	–	0.4593
GatedGCN [9]	104,217	117.63	941.04	**0.6731**
SplineCNN [22]	139,178	197 ± 82	–	0.5869
Spectral GNN				
H-L Cheby-net [17]	200,000	253*	63,756+	**0.7318**

method, which outperformed other methods based on different attributes in our experiments. Thus, the hierarchy-based, kNN-based, and complete-based are constructed from the hierarchy computed using watershed by area, which is applied to the RAG of the superpixels obtained by the SLIC.

5.1.2 Architecture

To implement our model, we use PyTorch Geometric [10], which batches multiple graphs into a single graph with multiple subgraphs for mini-batch training. Following [5], we could set the variable depth $\mathcal{M}$ as half of the number of vertices in each graph. However, this is not feasible for graphs with different sizes in the same batch. Instead, we use $\mathcal{M} = \lfloor max(|\mathcal{V}|_{minibatch})/2\rfloor$, in which $max(|\mathcal{V}|_{minibatch})$ is the maximum number of vertices among the graphs in the batch.

We use GatedGCN as the graph convolution network, with a slight modification of replacing batch normalization with layer normalization in Equations 3 and 5. This method preserves and updates the edge features w_{uv} between vertices u and v at each layer [9]. Moreover, the soft attention mechanism in Equation 5 enables the model to learn how important each neighbor v is for vertex u.

Distinct from other GCN models, this work used a readout layer that concatenates the global mean and the max pooling, capturing the average and maximum values of the vertex features from the entire graph. Combining two permutation invariant functions was motivated by the better results obtained in the initial experiments compared to the use of only one. It is worth mentioning that the MLP includes two linear layers with layer normalization that help improve the model's training stability and generalization performance. Table 1 shows the details of each layer in our proposed architecture.

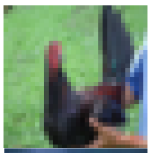
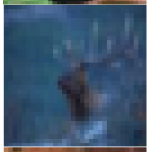
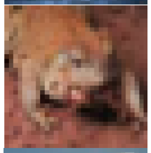
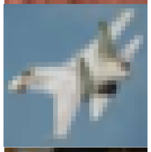
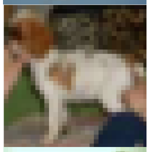
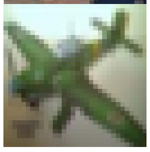
5.2 Quantitative analysis

Table 2 shows the results for the HGCIC model and other state-of-the-art methods. The HGCIC model was trained using three different graphs: kNN-based (HGCIC$_k$), hierarchy-based (HGCIC$_h$), and complete-based (HGCIC$_c$). Interestingly, the HGCIC$_h$ model, based on the hierarchy structure, has a much smaller number of edges and outperforms the other two graphs.

This result is noteworthy since the number of edges in a graph directly impacts the message-passing step of graph neural networks, which are designed to learn from the graph's structural information. However, our findings suggest that the relationships captured by edges in the graph are more critical than the graph's number of edges. This is consistent with prior work showing that incorporating hierarchical relationships between vertices and edges can help improve GNN performance [17]. Overall, test results demonstrate the effectiveness of the proposed hierarchical adjacency method in enhancing our model's performance.

Despite the superior performance of the HGCIC$_h$ model, our work did not achieve the best accuracy in image classification. However, we achieved a competitive result without resorting to complex strategies that other methods used, such as recurrent neural networks [5], multiple relations [17],

Table 3 Examples of predictions of the proposed models. Images have been enlarged for easy viewing.

	Groundtruth	Hierarchy-based	kNN-based	Complete-based
	Bird	Bird	Dog	Horse
	Deer	Deer	Bird	Ship
	Frog	Frog	Frog	Frog
	Airplane	Airplane	Airplane	Ship
	Dog	Dog	Bird	Dog
	Airplane	Bird	Frog	Frog

positional embeddings [21], or vector fields to define the directions of information propagation [3]. These techniques could also be incorporated into our model in the future to boost its performance further. Another factor that could affect our results is the size of our graphs, which were the smallest among the compared methods. The only model with fewer parameters than ours was the one proposed by [2]. Therefore, our work demonstrates a promising approach for graph-based image analysis and shows the potential of hierarchical segmentation for creating effective graph representations of images.

A drawback of the methods proposed in [17] is that they require computing the eigenvalues and eigenvectors of the graph Laplacian matrix. This step is essential for learning filters that depend on the Laplacian eigenbasis and capture the graph structural information, but it can be costly. Furthermore, since the Laplacian eigenbasis is specific to each graph, models trained on one graph may not generalize well to others. This limitation can restrict the scalability and applicability of these methods to a broader range of graph data. Using spatial GNNs, our model can capture the geometric features of the graph without relying on the Laplacian eigenbasis and avoid the problem of generalization to new graphs.

5.3 Qualitative analysis

In Table 3, we present the qualitative results of our models. We observe that the model trained with the hierarchy-based adjacency can classify not only simple images but also images that are challenging for humans to classify due to their low resolution (only 32×32).

5.4 Ablation Study

To showcase the performance improvement in using the chosen architecture, we conduct experiments with a similar architecture described in [9], consisting of four GatedGCN layers, a global average for the readout layer, and an MLP with two linear layers. Additionally, we tested our proposed

Table 4 Performance of the ablation study to assess how changes in the proposed method affect performance.

Model	# params	# vertices	# edges	Accuracy
Architecture proposed in [9]				
8-nn	145,136	47.13	377.04	0.4716
Hierarchy Adj	145,136	92.26	377.04	0.4658
Complete Graph	145,136	47.13	2,175.11	0.4803
Proposed architecture with RGB, X and Y features				
8-nn	90,278	47.13	377.04	0.546
Hierarchy Adj	90,278	47.13	92.26	0.548
Complete Graph	90,278	47.13	2,175.11	0.5698

graph creation process with a simple set of features as used in [2,9,17], which consist of the mean of RGB channels and X and Y mean position, to demonstrate that a more representative set of features also makes improvements in the task.

Table 4 shows the result of the models trained with the previously mentioned changes. All models showed degradation in accuracy, proving the benefits of the architecture and features used.

Out of all the models tested, the ones using architecture proposed in [9] showed the most significant decrease in accuracy. This was due to the limited capacity of the architecture to learn complex features from the data. Specifically, the four layers in this architecture were deemed insufficient for the graph-learned embeddings to converge, resulting in poor representations in the final GNN layer.

Among the models trained with the simplest features, the one with the highest accuracy was the model trained with complete graph adjacency. The superior performance of this model was because the distance between vertices is not a reliable indicator of segment differences. As a result, the model gave similar weightage to all neighbors in the message-passing, consequently prioritizing more neighbors to perform the feature aggregation. Similar behavior also occurred in experiments with the [9] architecture, where the hierarchy adjacency caused relevant vertices to be farther away from others. As previously said, the four layers are insufficient for the information to propagate throughout the graph.

6 Conclusion

This work introduces a new approach for constructing graphs from images using hierarchical segmentation methods. Additionally, it presents a new model called HGCIC, which has been trained using graphs obtained from hierarchical segmentation in three different adjacency setups. The proposed model has demonstrated remarkable results by incorporating variable depth, hierarchical relationships through edges, and well-defined features. The implications of this research are exciting and could have a far-reaching impact on various applications that rely on graph-image analysis.

Although the results were slightly inferior, they still showed promise. The proposed approach utilizes significantly smaller graphs than those in previous works, and the proposed GCN architecture contains fewer parameters yet still delivers promising results. Moreover, the ablative study confirmed the hypothesis that the choice of architecture and features positively impacted the model's overall performance. These findings highlight the potential of the proposed approach as a more efficient and effective means of graph construction and analysis.

In future works, we plan to investigate the impact of attention mechanisms on our approach. Additionally, we aim to conduct a more in-depth analysis of the relationship between the number of vertices and model performance while exploring multiple graph relations.

Acknowledgements

Omitted for double-blind review.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE transactions on pattern analysis and machine intelligence **34**(11), 2274–2282 (2012)
2. Avelar, P.H., Tavares, A.R., da Silveira, T.L., Jung, C.R., Lamb, L.C.: Superpixel image classification with graph attention networks. In: 2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 203–209. IEEE (2020)
3. Beaini, D., Passaro, S., Létourneau, V., Hamilton, W., Corso, G., Liò, P.: Directional graph networks. In: International Conference on Machine Learning, pp. 748–758. PMLR (2021)
4. Bresson, X., Laurent, T.: Residual gated graph convnets. arXiv preprint arXiv:1711.07553 (2017)
5. Corso, G., Cavalleri, L., Beaini, D., Liò, P., Veličković, P.: Principal neighbourhood aggregation for graph nets. Advances in Neural Information Processing Systems **33**, 13,260–13,271 (2020)
6. Cousty, J., Najman, L.: Incremental algorithm for hierarchical minimum spanning forests and saliency of watershed cuts. In: International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing, pp. 272–283. Springer (2011)

7. Cousty, J., Najman, L., Kenmochi, Y., Guimarães, S.: Hierarchical segmentations with graphs: Quasi-flat zones, minimum spanning trees, and saliency maps. J. Math. Imaging Vis. **60**(4), 479–502 (2018)
8. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems **29** (2016)
9. Dwivedi, V.P., Joshi, C.K., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Benchmarking graph neural networks. Journal of Machine Learning Research **24**(43), 1–48 (2023)
10. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds (2019)
11. Fey, M., Lenssen, J.E., Weichert, F., Müller, H.: Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 869–877 (2018)
12. Guigues, L., Cocquerez, J.P., Le Men, H.: Scale-sets image analysis. International Journal of Computer Vision **68**(3), 289–317 (2006)
13. Guimarães, S., Kenmochi, Y., Cousty, J., Patrocinio, Z., Najman, L.: Hierarchizing graph-based image segmentation algorithms relying on region dissimilarity. Mathematical Morphology-Theory and Applications **2**(1), 55–75 (2017)
14. Han, K., Wang, Y., Guo, J., Tang, Y., Wu, E.: Vision gnn: An image is worth graph of nodes. In: NeurIPS (2022)
15. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning, pp. 448–456. PMLR (2015)
16. Johnson, J., Krishna, R., Stark, M., Li, L.J., Shamma, D., Bernstein, M., Fei-Fei, L.: Image retrieval using scene graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3668–3678 (2015)
17. Knyazev, B., Lin, X., Amer, M., Taylor, G.: Image classification with hierarchical multigraph networks. In: K. Sidorov, Y. Hicks (eds.) Proceedings of the British Machine Vision Conference (BMVC), pp. 223.1–223.13. BMVA Press (2019). DOI 10.5244/C.33.223. URL `https://dx.doi.org/10.5244/C.33.223`
18. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
19. Liu, Q., Xiao, L., Yang, J., Wei, Z.: Cnn-enhanced graph convolutional network with pixel-and superpixel-level feature fusion for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing **59**(10), 8657–8671 (2020)
20. Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., Bronstein, M.M.: Geometric deep learning on graphs and manifolds using mixture model cnns. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5115–5124 (2017)
21. Rampášek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G., Beaini, D.: Recipe for a general, powerful, scalable graph transformer. Advances in Neural Information Processing Systems **35**, 14,501–14,515 (2022)
22. Vasudevan, V., Bassenne, M., Islam, M.T., Xing, L.: Image classification using graph neural network and multiscale wavelet superpixels. Pattern Recognition Letters (2023)
23. Wan, S., Gong, C., Zhong, P., Du, B., Zhang, L., Yang, J.: Multiscale dynamic graph convolutional network for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing **58**(5), 3162–3177 (2019)
24. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.: A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems (2020)