Graph-based Multibeam Forward Looking Acoustic Image Classification

Gabriel Arruda Evangelista[1], **João Baptista de Oliveira e Souza Filho**[1]

[1] Programa de Engenharia Elétrica / COPPE
Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro – RJ – Brazil

{gabrielevangelista7, jbfilho}@poli.ufrj.br

Abstract. *Multibeam sonar imaging has many applications, such as mine-like detection and navigation tasks, motivating interest in the automatic classification of sonar images. Recent works have proposed graph neural networks (GNNs) as an alternative to convolutional neural networks (CNNs) to address this task. This paper focuses on combining the strengths of both models to enhance the performance of GNNs when classifying sonar images. This proposal exploits a superpixel algorithm for image segmentation and graph formation. Comprehensive experiments with an MFLS open dataset evaluate the effect of model design parameters on the performance of the proposed approach. Using CNN-extracted features as initial node embeddings significantly improved the graph-based image classification performance.*

1. Introduction

Over the past few decades, sonar systems have played a crucial role in remote sensing of underwater environments, providing valuable insights about bodies of water and the underlying seabed. The advance of sonar technologies has enabled the development of sophisticated imaging techniques in this context, like Side Scan Sonar (SSS) and Multibeam Forward Looking Sonar (MFLS), facilitating detailed imaging and mapping of the seafloor and surrounding structures. The popularisation of such systems has enabled many recent applications, such as bathymetric surveys, employed in autonomous underwater navigation systems [Dos Santos et al. 2022, Galceran et al. 2012], as well as the inspection and search of objects located on the seafloor, such as mines or mine-like objects (MLOs), which can pose navigation hazards [Sinai et al. 2016].

Given the increasing importance of Computer Vision (CV) techniques in recent technologies, like autonomous and automatic identification systems, there is a rising interest in their application to sonar images [Steiniger et al. 2022]. Some use cases include underwater object classification [Ye et al. 2018, Huo et al. 2020], semantic segmentation for seabed surface classification [Yang et al. 2022], and object detection [Yu et al. 2021]. However, the expensive and secure nature of collecting sonar data for a given task implies the scarcity of open datasets. [Singh and Valdenegro-Toro 2021] introduced a dataset specifically designed for semantic segmentation of marine debris using MFLS images. Similarly, [Xie et al. 2022] presented a dataset comprising images of some objects, primarily targeting the object detection task.

According to the survey conducted by [Steiniger et al. 2022], most works since 2016 have embraced deep learning models, particularly the Convolutional Neural Networks (CNNs) when dealing with image classification tasks. However, in some alternative CV domains, there is an increasing number of initiatives involving graph-based

models [Vasudevan et al. 2023, Avelar et al. 2020, Dwivedi et al. 2023, Knyazev et al. 2019]. In line with this, the present work addresses the problem of classifying an open-source dataset of sonar images exploiting Graph-Neural Networks (GNNs). We propose a novel method that combines the strengths of both CNNs and GNNs in image classification tasks. Mainly, our proposal exploits the CNN model to produce initial node embeddings refined by the GNNs, leading to expressive gains in the classification performance but not surpassing a standard CNN classification model.

This paper is organized as follows: Section 2 introduces the proposed hybrid CNN-GCN model for image classification. Section 3 describes the open-source MFLS image dataset used to evaluate the proposed model. Section 4 outlines the experimental approach for assessing model performance. Results are presented in Section 5. Finally, Section 6 gives the conclusions.

2. Methodology

In general, CNN is the standard approach for image classification. However, with the recent advances in Graph Neural Networks, studies are attempting to solve image classification problems using GNNs. In the literature, some researchers [Vasudevan et al. 2023, Avelar et al. 2020, Dwivedi et al. 2023, Knyazev et al. 2019] have been tackling the issue of converting an image into a graph by employing superpixels generated by algorithms such as the SLIC [Achanta et al. 2012].

Superpixels are image segments grouping adjacent pixels which share similar characteristics. Representing an image using superpixels provides meaningful semantic and structural region identification. Therefore, a natural assumption is associating each superpixel with a graph node when representing an image by a graph.

In these graphs, for establishing the node links, i.e., modeling the relations between the superpixels, [Belkin and Niyogi 2001] originally proposed the inclusion of only the k nearest neighbour nodes (kNN), whose edge weights are determined by Eq. (1), where $\mathbf{x_i}$ represents the vector of attributes related to the ith node, and t is a user-defined temperature parameter.

$$w_{ij} = \exp\left(-\frac{\|\mathbf{x_i} - \mathbf{x_j}\|^2}{t}\right) \tag{1}$$

Other works such as [Avelar et al. 2020, Dwivedi et al. 2023, Knyazev et al. 2019] employ the normalized distance between the superpixel centroids to define the edge weights. Alternatively, [Lei et al. 2022] proposes to combine the geometric distance between nodes and their attributes. In this work, we used the edge weights only for establishing node links via the kNN algorithm since the evaluated GNN models do not consider this graph attribute.

2.1. Convolutional Neural Networks

Neural network models that exploit convolutional layers are specially tailored for grid-like data and have been showing distinguishing performance in many CV tasks. Each CNN layer includes multiple convolutional filters which capture meaningful features from the input images, such as textures or edges, through networking training. Consequently, after

applying a convolutional layer, image regions are represented by multiple attributes resulting from these filters, named feature maps. Another standard structure that CNNs explore is the pooling layer, which condenses such feature mappings into smaller and denser representations. In summary, these layers aggregate the information initially embedded into a few image channels of high-dimensional maps into progressively smaller maps having more channels. This leads to a reduced number of complex and better problem-descriptive feature vectors.

In the sonar image context, in [Xie et al. 2022], some preliminary image segmentation results accompanied by object classification are discussed. Faster R-CNN [Ren et al. 2015] was the best-performing model, using the ResNet-18 as the backbone network. ResNet-18 belongs to the Residual Neural Network (ResNet) family [He et al. 2016], an architecture designed to mitigate the vanishing gradient problem in deep neural networks.

2.2. Graph Neural Networks

Graph Neural Networks (GNNs) are architectures designed to handle graph-structured data. In this case, the data is modeled using entities represented by graph nodes and their relationships, typically sparse, exploring edges. The learning tasks over a graph usually focus on inferring information about the nodes, edges, or the entire graph. Some GNN layers have the task of performing graph convolution, an operation that implements a message-passing mechanism among nodes and their neighbours. This means that each node embedding will aggregate information from the node as those carried by the features assigned to its neighbour nodes. Each layer added to a GNN model increases this neighbourhood coverage. The dimensionality of each GNN layer output is referred to as the number of "hidden channels" or the "hidden dimensions" and represents a design parameter. The following GNN models were considered in our experiments:

1. The Graph Convolutional Network (GCN) model was proposed in [Kipf and Welling 2017] for semi-supervised graph node classification. This model employs message-passing techniques based on the relational structure among nodes, utilizing graph convolutions to capture local contexts effectively.
2. The Graph Attention Networks (GAT) model proposed in [Veličković et al. 2018] explores the attention mechanism [Vaswani et al. 2017] to calculate the relative importance of the neighbourhood information adaptively. This strategy is implemented by attention heads operating in parallel to stabilize the learning process.
3. The Graph Isomorphism Network (GIN) model proposed in [Xu et al. 2019] aims to achieve an injective representation of node graph embeddings to ensure better discriminability over different graph structures. For this purpose, the model employs two Multilayer Perceptrons (MLPs).

2.3. A hybrid CNN-GCN model for image classification

Recent studies have introduced GNN-based models in CV problems involving non-traditional images and geometric structures, such as point clouds [Mitrokhin et al. 2020, Schaefer et al. 2022, Ding et al. 2022, Shi and Rajkumar 2020, Lei et al. 2022]. MFLSs produce 2D greyscale images reflecting 3D scenes, wherein each pixel intensity corresponds to the "strength" of the received signal, carrying information about the object's distance, the sea's acoustic conditions, and the target's material properties.

To tackle this problem, our model tries to benefit from the strengths of both grid-like representations, where the CNNs excel, and the geometric aspects of graph-represented data, to which GNNs have shown high potential. Superpixels serve as a common mediator between images and graphs, the latter enabling the exploration of associations among non-neighbour regions of an image.

The proposed framework is outlined in Figure 1, where the input is one preprocessed image, represented by a tensor of dimensions $(C \times H \times W)$, denoting the number of channels, width, and height, respectively. The critical components depicted in this diagram include:

- SLIC [Achanta et al. 2012] (Segmentation Algorithm): the superpixel algorithm operates on the preprocessed image to obtain a mask with the same dimensions as the input, where each pixel is labeled with the corresponding superpixel (node) index. This process results in a superpixels' mask and a matrix of distances between superpixels' centroids.
- Resize: pre-trained CNNs architectures typically assume square images as inputs, thus have input tensors with dimensions given by $(C_{CNNin} \times W_{CNNin} \times W_{CNNin})$, i.e., $H_{CNNin} = W_{CNNin}$. Thus, rectangular images are resized by the nearest pixel resampling approach, such that the larger image dimension must match W_{CNNin}, the aspect ratio is preserved, and the resulting image is centered into the target square shape by padding with zeros. The superpixel mask undergoes the same process.
- Mask Projection: since each pixel mask is associated with a superpixel (node) index, this mask must be resized to match the CNN feature extractor map size (described in the following) in a similar fashion to the region projection mapping explored by the Fast R-CNN [Girshick 2015].
- Partial CNN (Backbone Network): this network aims to enrich initial node representations with meaningful image attributes. This task considers part of the processing stages involved in a standard convolutional-based image classification model, i.e., it adopts a cropped neural model such that all layers after some user-defined structural cutting point are removed. Thus, this block assumes as an input a tensor with the standard shape of the chosen backbone, i.e., $C_{CNNin} \times W_{CNNin} \times W_{CNNin}$. Defined a cutting point, the corresponding feature map generated at the block output will have dimensions $F_{CNNout} \times W_{CNNout} \times W_{CNNout}$, leading to W^2_{CNNout} feature vectors with dimensionality F_{CNNout} to be explored in downstream tasks. Typically, the feature map size is smaller than that of the network input, as $W_{CNNout} < W_{CNNin}$, but each feature vector has more dimensions since $F_{CNNout} > C_{CNNin}$, thus it is more problem representative. Our experiments adopted the default pretrained ResNet18 [He et al. 2016] model, available in the torchvision Python library (IMAGENET1K_V1), trained for the ImageNet 1000 classes image classification task [Deng et al. 2009].
- Edge forming: this process exploits the kNN approach, described in Section 2, assuming edges' weights are defined by the normalized distances between the superpixels' centroids.
- Node Feature Pooling: due to the correspondence assumed between each superpixel and a graph node, based on the mask projection, the feature map positions corresponding to the same superpixel are aggregated into a single feature vector by

averaging. Reducing the feature map size related to the CNN's cropping process may lead to some superpixels not being mapped. The unmapped nodes assume the average values of the feature vectors from their one-hop neighbour nodes.

- GNN: it has the task of refining the initial node embeddings produced in the previous stage. The corresponding final embedding assigned to each node is then pooled to create a single vector representing the entire graph.
- Classification: Based on the graph feature vector, this stage exploits a softmax layer to estimate the likelihood of each class. The final class assignment is based on the MAP criterium.

3. Dataset

The data explored in our experiments were sampled from a dataset created for the object detection task described in [Xie et al. 2022]. The greyscale images were captured by a Gemini 1200ik model MFLS (Multibeam Forward Looking Sonar) using frequencies of 720 and 1200kHz. The sonar system features a horizontal beam aperture spanning 120 degrees and a vertical aperture of 20 degrees for higher frequencies and 12 degrees for lower frequencies. These frequencies correspond to an angular resolution ranging from 0.25 to 0.12 degrees, yielding a range resolution from 2.4 to 4 millimeters. The pixel intensity in this sonar image is intrinsically linked to the amplitude of the echo of the emitted pulse. Many factors influence the value of this amplitude, including the source level, grazing angle, target composition, texture, and geometric configuration. Moreover, the signal return time is intrinsically tied to the target's distance. [Bjørnø et al. 2017]. The dataset images cover objects with dimensions (widths, lengths, and radius) ranging from 0.2 to 1.5 meters, positioned at distances from 5 to 25 meters to the sensor. The samples were made available and split into three partitions originally referred to as "Training", "Test_1", and "Test_2". The partition initially designated as "Test_1" was considered the validation set, while "Test_2" defined the test set.

The original images were partitioned into sub-images containing just one object and had the corresponding object label assigned. As a result, the resulting dataset disposes of 14,639 object images with various sizes, distributed across ten classes, as presented in Table 1. Following the same distribution observed in the original subsets, these sub-images were split among training, validation, and testing sets.

Table 1. Distribution of dataset instances per class and partition.

Class\Partition	Training	Test 1	Test 2	Total
ball	3072	197	193	3462
circle cage	661	99	99	859
cube	2644	172	168	2984
cylinder	564	48	45	657
human body	1281	76	73	1430
metal bucket	476	6	5	487
plane	795	134	135	1064
rov	700	150	150	1000
square cage	980	167	168	1315
tyre	1123	121	121	1365

Figure 2 shows the histogram of pixel intensities per class, partition, and overall. The class-wise histogram reveals variations in the maximum pixel intensity among the dif-

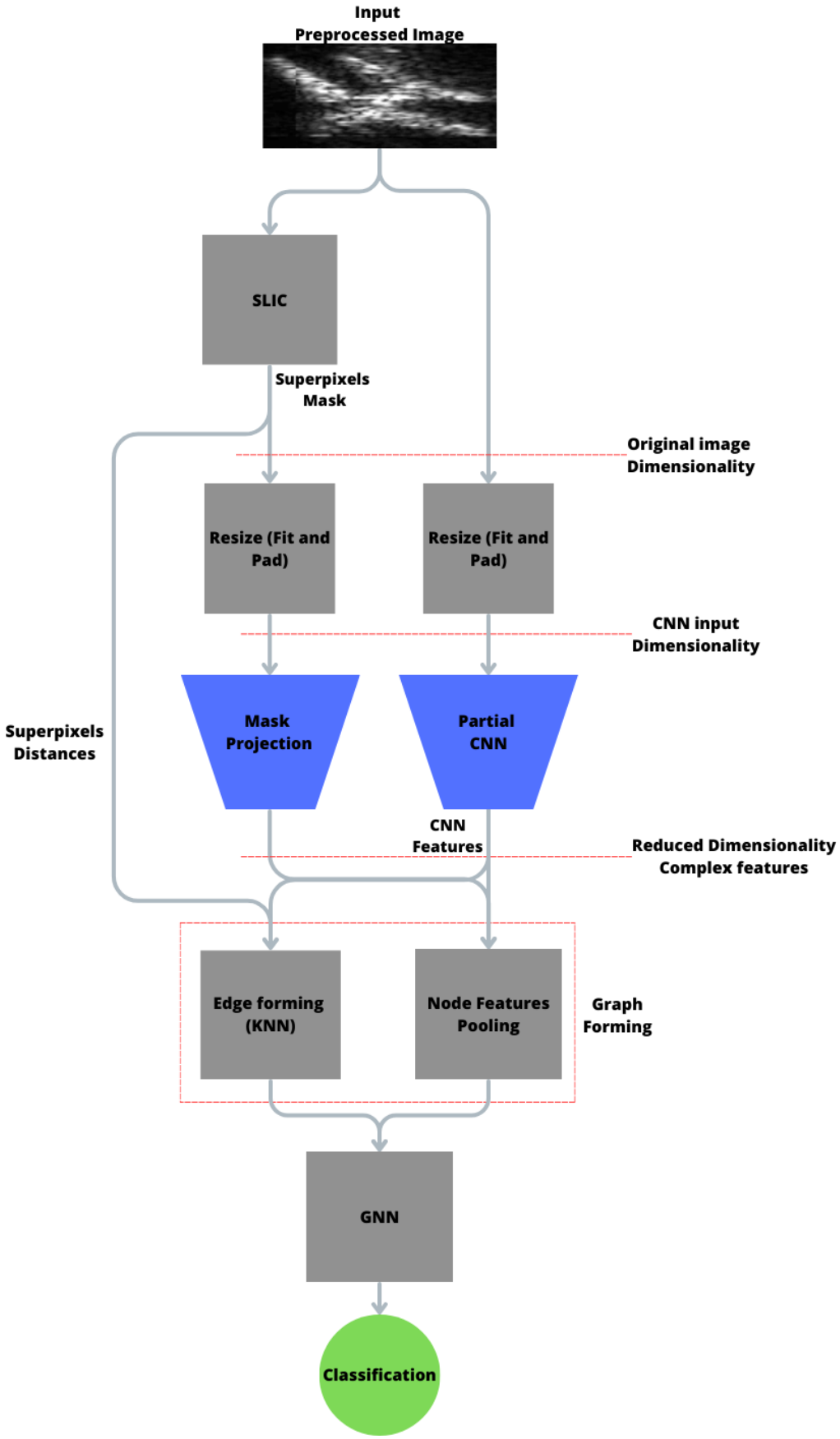

Figure 1. Illustrative diagram of the proposed method.

ferent classes. The overlap observed in the partition-wise histograms is high, suggesting a good statistical matching between the instances from the training, validation, and test sets. However, the overall histogram demonstrates a noticeable bias toward intensities below 50 on a scale from 0 to 255, signaling very dark images. Therefore, we considered incorporating histogram-based preprocessing techniques to enhance image contrast. Given that, the following schemes were considered for preprocessing the original image:

1. Original: no processing is conducted;
2. Auto contrast: here, the pixel intensities are normalized, disregarding those below and above the quantiles 0.5% and 99.5%, respectively.
3. Equalize: it explores a non-linear mapping to create a new image wherein the pixel intensities are uniformly distributed.

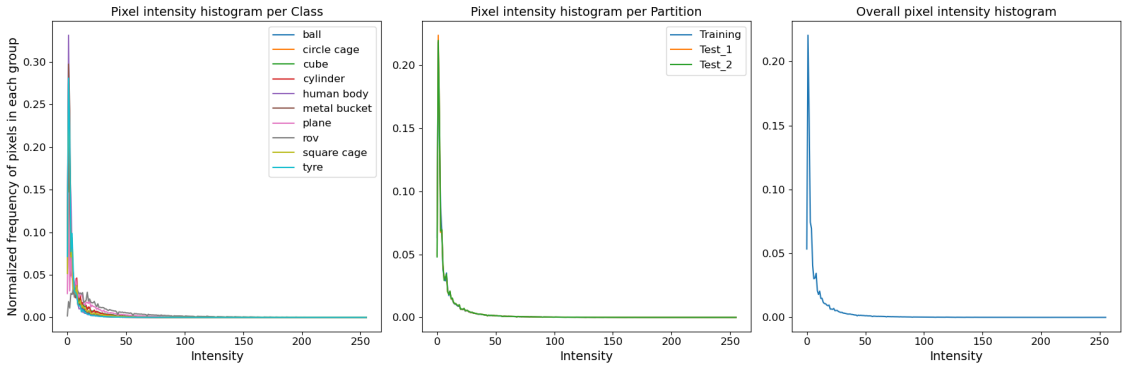

Figure 2. Histogram of pixel intensities stratified per class, partition, and overall (see text).

4. Experiments

The effectiveness of the proposed method for classifying MFLS-generated images was evaluated through several numerical experiments. The SLIC algorithm was settled to generate approximately 75 regions (nodes) in all experiments, as in [Avelar et al. 2020, Dwivedi et al. 2023]. All models were implemented using the PReLU activation function following the results presented in [You et al. 2020]. The GNN's hyperparameters considered in our experiments were *hidden_dim*, *num_layers*, and *pooling method*. The hidden dimension parameter (*hidden_dim*), which refers to the dimensionality of the hidden message-passing layer, was set to the same value for all GNN layers. The *num_layers* hyperparameter represents the number of message-passing layers in each model. In turn, the graph pooling strategy defines the process of mapping all node features into a single graph feature vector. A softmax layer was used for feature vector graph classification in all models. Table 2 summarizes the range of hyperparameters considered in these experiments. All GAT models considered four attention heads in the message-passing layers. The MLPs networks explored by the GIN models adopted two hidden layers, with the number of neurons defined by the hyperparameter *hidden_dim*.

4.1. Experiment 1 - GNN hyperparameters choice for standard graph-based image classification

The first experiment considered the standard procedure used for graph-based image classification, adopting the average pixel intensity over each superpixel region as the initial

Table 2. Range of hyperparameters evaluated in Experiment 1.

Parameter	Values
Convolutional Layer	{GCN; GAT; GIN}
Hidden Dimension	{8; 16; 32}
Num. Layers	{2; 4; 8}
Graph Pooling Layer	{global_mean_pool; global_add_pool}
Preprocessing	{original; autocontrast; equalize}

node embedding [Dwivedi et al. 2023], instead of the values provided by the CNN-based feature extractor discussed in Section 2.3. The grid-search procedure considered the range of hyperparameters reported in Table 2. The performance metric was the average validation accuracy. The training set accuracy was also monitored to identify a possible occurrence of overtraining. This experiment aimed to establish a baseline for the subsequent evaluations of the proposed model. Besides, it helped guide the choice of the parameters considered in the upcoming experiments.

4.2. Experiment 2 - Hybrid CNN-GNN proposed model

The second experiment evaluated the proposed method according to the same metrics adopted in Experiment 1. Here, we also investigated the influence of the CNN cutting point. This evaluation process was conducted in a greedy fashion, establishing the feature map of the first CNN layer as the cutting point 1. At each stage, one or more layers were added to the network structure to define a new cutting point. As previously discussed, this cutting point affects both the feature map size and the dimensionality of the node features to be subsequently processed by the GNN. Since the cutting point settlement involves conflicting factors, this experiment focuses on determining proper design values and which factors are more influential to our task. Table 3 summarizes the different cutting points evaluated, the corresponding feature map sizes, and the dimensionality of the resulting feature maps. In this manner, the second experiment is specifically designed to directly compare the proposed model and other models that utilize GNNs for image classification. We also explore the influence of the deepness of the CNN-based extractor.

Table 3. Layers added to the CNN feature generator for each cutting point settled (see text).

Cutting point	Added layer(s)	Feature vector dimension	Feature map size
1	Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3))	64	112, 112
2	BatchNorm2d(64, eps=1e-05, momentum=0.1)	64	112, 112
5	ReLU MaxPool2d(kernel_size=3, stride=2, padding=1, dilation=1) BasicBlockPair(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))	64	56, 56
8	BasicBlockPair(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1)) BasicBlockPair(256, 512, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))	512	7, 7

The range of hyperparameters considered in these experiments was defined with basis on the results of Experiment 1, discussed in Section 5, and summarized in Table 4.

4.3. Experiment 3 - CNN feature extraction module fine-tuning

The last experiment evaluated the impact of fine-tuning the CNN-based feature extraction module, targeting the classification of sonar images. It considered the multiple cuts in Table 4. For this purpose, the images were resized by the process described in Section 2.3.

Table 4. Range of hyperparameters evaluated in Experiment 2.

Parameter	Values
Convolutional Layer	{GCN; GAT}
Hidden Dimension	{8; 32}
Num. Layers	{4; 8}
Graph Pooling Layer	{global_mean_pool}
Preprocessing	{original; autocontrast}
Cut	{1; 2; 5; 8}

5. Results

In the following, the results of the previously described experiments are reported.

5.1. Experiment 1

The validation accuracy of each model is depicted using a boxplot graph in Figure 3. The three best-performing models for each type of message-passing layer are described in Table 5. Figure 4 depicts boxplot graphs of models' accuracy associated with different preprocessing schemes and hyperparameter choices. Table 6 consolidates the top five results considering all hyperparameters' combinations.

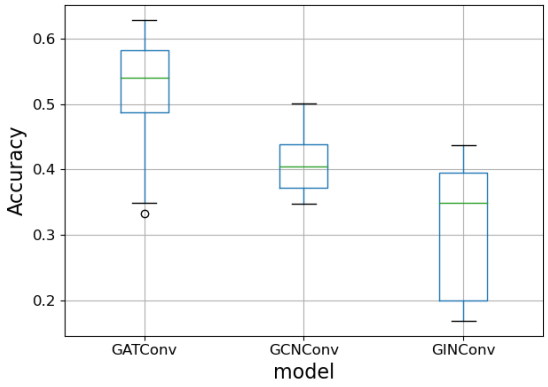

Figure 3. Boxplot graph for the validation accuracy per model.

Table 5. Top-three models per message passing modality for Experiment 1.

Model	n_layers	Hidden_channels	Pooling	Preprocessing	Validation Accuracy
GATConv	4	8	global_mean_pool	autocontrast1	0.6291
				equalize	0.6274
		32	global_mean_pool	equalize	0.6274
GCNConv	4	32	global_mean_pool	autocontrast1	0.5017
	8	32	global_mean_pool	original	0.4991
				autocontrast1	0.4974
GINConv	2	8	global_mean_pool	autocontrast1	0.4368
		16	global_mean_pool	autocontrast1	0.4368
		32	global_mean_pool	autocontrast1	0.4333

One may note that models employing GAT layers outperformed all the alternative models. Global mean pooling was the best choice for all the top five models. Furthermore, based on Figure 4, GNN models with four layers performed slightly better. The differences observed were minimal regarding the *hidden_channels* hyperparameter. Finally, the

Table 6. Hyperparameters related to the top five models evaluated in Experiment 1.

Model	n_layers	Hidden_channels	Pooling	Preprocessing	Best_epoch	Training Accuracy	Validation Accuracy
GATConv	4	8	global_mean_pool	autocontrast1	219	0.7830	0.6291
				equalize	97	0.7459	0.6274
		32	global_mean_pool	equalize	196	0.7785	0.6274
		16	global_mean_pool	original	257	0.7639	0.6231
		8	global_mean_pool	original	202	0.7506	0.6188

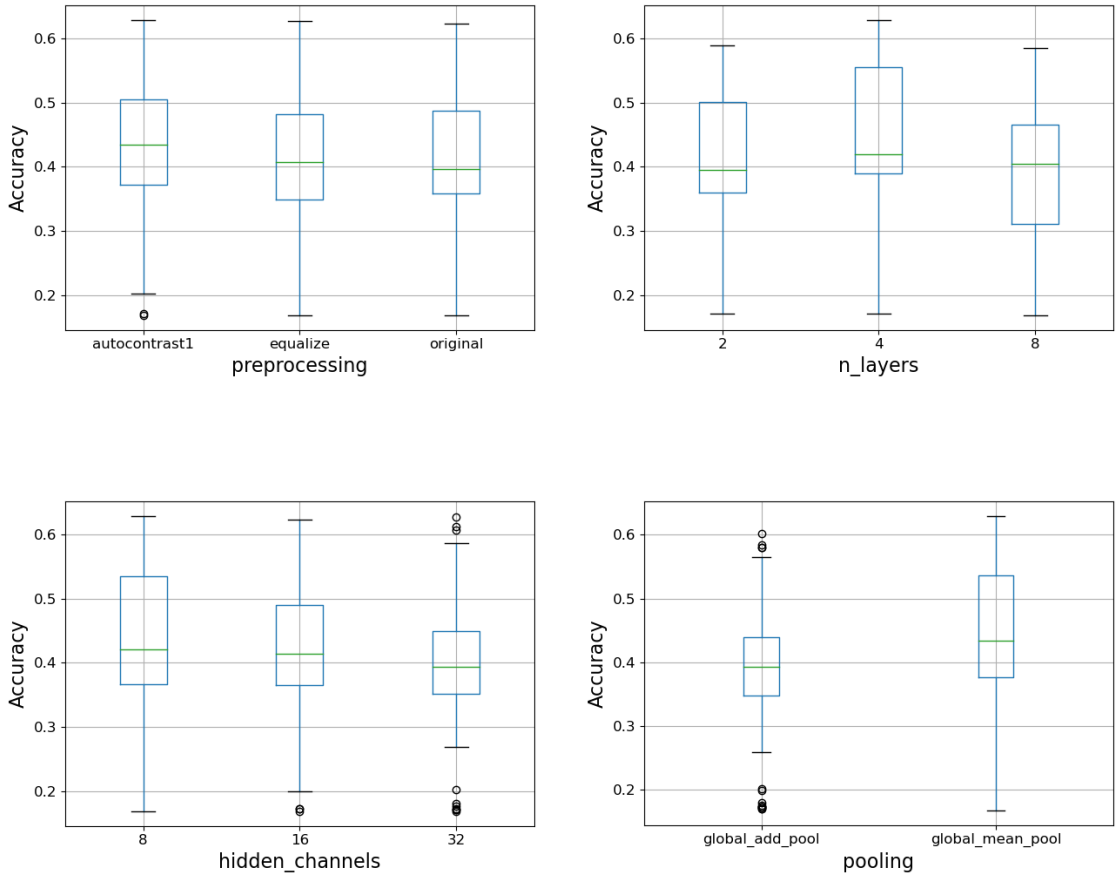

Figure 4. Boxplot graphs for the models' accuracy considering different preprocessing schemes and hyperparameters' choices.

global_mean_pooling achieved a higher median value than the *global_add_pooling* strategy.

5.2. Experiment 2

This experiment focused on the proposed mixed CNN-GNN model, described in Section 2. Table 7 reports the five best-performing models in this experiment. The CNN-based feature extractor effectively generated the initial node embeddings, leading to a higher validation accuracy of almost 14 points compared to the baseline model (Experiment 1). However, the differences between the training and validation accuracies observed in Table 7 may result from overfitting, an aspect to further investigate in future works.

Table 7. Hyperparameters and some performance indexes for the top-five models identified in Experiment 2.

Model	n_layers	Hidden_channels	Preprocessing	Cutting point	Best_epoch	Training Accuracy	Validation Accuracy
GCNConv	4	32	original	8	126	0.9835	0.7641
	8	32	original	8	137	0.9474	0.7556
	4	32	autocontrast1	8	101	0.9983	0.7538
GATConv	4	32	autocontrast1	8	78	0.9687	0.7479
		8	original	8	110	0.9819	0.7462

Figure 5 analyses the effects on the overall classification performance of setting different cutting points in the CNN-based feature extractor. One may observe an increasing trend in accuracy with the rise in the number of CNN layers. Table 8 resumes the best-performing models identified at each cutting point. Feature dimensionality has shown to be a crucial design factor, predominantly affecting model performance. The increase in the feature vector dimensionality also seems to compensate for the possible effects of reducing the feature map size, thus, the number of superpixels explored by the proposed algorithm.

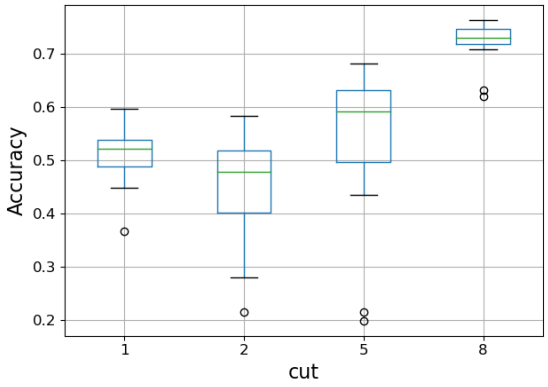

Figure 5. Boxplot graph of the models' validation accuracy, considering different cutting points for the CNN feature extractor (see text).

5.3. Experiment 3

The third experiment analyzed the influence of fine-tuning or not the feature extraction model for the original dataset task (i.e., for sonar image classification). Figure 6 depicts the boxplot graph associated with this experiment. Surprisingly, this fine-tuning process has not led to a significant increase in the models' performance.

Table 8. The top-two models for each CNN cutting point considered in Experiment 2.

Cutting point	Model	n_layers	Hidden_channels	pooling	Preprocessing	Validation Accuracy
1	GATConv	4	8	global_mean_pool	autocontrast1	0.5966
					original	0.5821
2	GATConv	4	32	global_mean_pool	autocontrast1	0.5829
			8	global_mean_pool	original	0.5436
5	GATConv	4	8	global_mean_pool	autocontrast1	0.6829
	GCNConv	4	32	global_mean_pool	autocontrast1	0.6803
8	GCNConv	4	32	global_mean_pool	original	0.7641
		8	32	global_mean_pool	original	0.7556

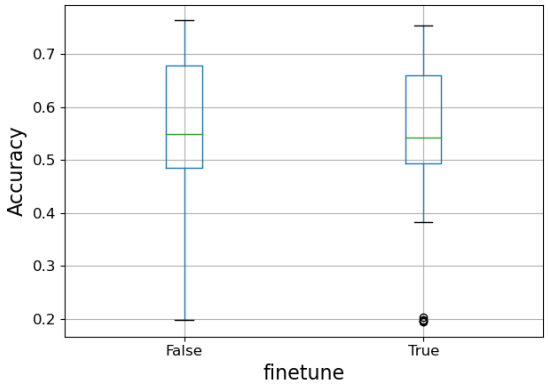

Figure 6. Boxplot graph for the validation accuracy obtained without or with fine-tuning the CNN feature extractor (see text).

5.4. Overall models' comparison

Table 9 points out the best-performing architecture identified in each experiment. Table 10 reports the recall values associated with each object class. Here, all models considered the *global_mean_pooling*. When compared to [Xie et al. 2022], wherein the ResNet-18 was used as the backbone for the Faster-RCNN [Ren et al. 2015] to object localization and classification, our results are somewhat optimistic, as our model restricts to the classification stage. Nevertheless, the proposed method vastly outperforms the standard graph-based image classification approach considered in Experiment 1.

Table 9. The best-performing models identified in each experiment.

Experiment \Parameters	layer type	num_layer	hidden_dim	preprocessing	Features	CNN cut	CNN finetune
Experiment 1	GAT	4	8	autocontrast	Pixel intensity	-	-
Experiment 2	GCN	4	32	original	Partial CNN	8	False
Experiment 3	GCN	4	32	original	Partial CNN	8	True

Table 10. Recall values and their average (mAR) for the test set, considering the models from Table 9, and the mAR value reported in [Xie et al. 2022] (see text).

	mAR	AR_{ball}	AR_{cc}	AR_{cube}	AR_{cy}	AR_{hb}	AR_{mb}	AR_{plane}	AR_{rov}	AR_{sc}	AR_{tyre}
Experiment 1	0.561	0.751	0.192	0.613	0.311	0.685	0.000	0.541	0.920	0.304,	0.463
Experiment 2	0.704	0.829	0.303	0.655	0.554	0.836	1.000	0.985	0.920	0.369	0.752
Experiment 3	0.715	0.891	0.232	0.673	0.511	0.849	1.000	0.985	0.913	0.429	0.719
ResNet18 [Xie et al. 2022]	0.897	-	-	-	-	-	-	-	-	-	-

6. Conclusion

This study introduced an enhanced graph-based image classification approach by combining CNNs and GNNs. The proposed model explores a superpixel algorithm to produce a graph representation of the input image. Over this graph, initial node embeddings are produced by a cropped CNN and subsequently refined by a GNN. Then, the model aggregated the resulting node features to produce a graph embedding vector exploited for image classification. This process combines the strengths of both CNN and GNN synergistically and may leverage the classification performance of input images by exploring associations between non-neighbours image regions.

Experiments conducted with a dataset of MFLS sonar images aimed to evaluate the effects on the performance of each design parameter. The experimental results confirm that the proposed model outperforms standard graph-based methods but still falls short of a standard CNN model in sonar image classification.

7. Acknowledgments

To CNPq, FAPERJ, and CAPES. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282.

Avelar, P. H., Tavares, A. R., da Silveira, T. L., Jung, C. R., and Lamb, L. C. (2020). Superpixel image classification with graph attention networks. In *2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 203–209. IEEE.

Belkin, M. and Niyogi, P. (2001). Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 14.

Bjørnø, L., Neighbors, T., and Bradley, D. (2017). *Applied underwater acoustics*. Elsevier.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE.

Ding, Y., Zhang, Z., Zhao, X., Hong, D., Cai, W., Yu, C., Yang, N., and Cai, W. (2022). Multi-feature fusion: graph neural network and CNN combining for hyperspectral image classification. *Neurocomputing*, 501:246–257.

Dos Santos, M. M., De Giacomo, G. G., Drews-Jr, P. L., and Botelho, S. S. (2022). Cross-view and cross-domain underwater localization based on optical aerial and acoustic underwater images. *IEEE Robotics and Automation Letters*, 7(2):4969–4974.

Dwivedi, V. P., Joshi, C. K., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. (2023). Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48.

Galceran, E., Djapic, V., Carreras, M., and Williams, D. P. (2012). A real-time underwater object detection algorithm for multi-beam forward looking sonar. *IFAC Proceedings Volumes*, 45(5):306–311.

Girshick, R. (2015). Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Huo, G., Wu, Z., and Li, J. (2020). Underwater object classification in sidescan sonar images using deep transfer learning and semisynthetic training data. *IEEE access*, 8:47407–47418.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Knyazev, B., Taylor, G. W., and Amer, M. (2019). Understanding attention and generalization in graph neural networks. *Advances in neural information processing systems*, 32.

Lei, C., Wang, H., and Lei, J. (2022). SI-GAT: A method based on improved graph attention network for sonar image classification. *arXiv preprint arXiv:2211.15133*.

Mitrokhin, A., Hua, Z., Fermuller, C., and Aloimonos, Y. (2020). Learning visual motion segmentation using event surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14414–14423.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Schaefer, S., Gehrig, D., and Scaramuzza, D. (2022). AEGNN: asynchronous event-based graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12371–12381.

Shi, W. and Rajkumar, R. (2020). Point-GNN: graph neural network for 3D object detection in a point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719.

Sinai, A., Amar, A., and Gilboa, G. (2016). Mine-like objects detection in side-scan sonar images using a shadows-highlights geometrical features space. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–6. IEEE.

Singh, D. and Valdenegro-Toro, M. (2021). The marine debris dataset for forward-looking sonar semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3741–3749.

Steiniger, Y., Kraus, D., and Meisen, T. (2022). Survey on deep learning based computer vision for sonar imagery. *Engineering Applications of Artificial Intelligence*, 114:105157.

Vasudevan, V., Bassenne, M., Islam, M. T., and Xing, L. (2023). Image classification using graph neural network and multiscale wavelet superpixels. *Pattern Recognition Letters*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2018). Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

Xie, K., Yang, J., and Qiu, K. (2022). A dataset with multibeam forward-looking sonar for underwater object detection. *Scientific Data*, 9(1):739.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019). How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*.

Yang, D., Cheng, C., Wang, C., Pan, G., and Zhang, F. (2022). Side-scan sonar image segmentation based on multi-channel CNN for AUV navigation. *Frontiers in Neurorobotics*, 16:928206.

Ye, X., Li, C., Zhang, S., Yang, P., and Li, X. (2018). Research on side-scan sonar image target classification method based on transfer learning. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–6. IEEE.

You, J., Ying, Z., and Leskovec, J. (2020). Design space for graph neural networks. *Advances in Neural Information Processing Systems*, 33:17009–17021.

Yu, Y., Zhao, J., Gong, Q., Huang, C., Zheng, G., and Ma, J. (2021). Real-time underwater maritime object detection in side-scan sonar images based on Transformer-YOLOv5. *Remote Sensing*, 13(18):3555.