sensors

MDPI

Article

Graph-Based Audio Classification Using Pre-Trained Models and Graph Neural Networks

Andrés Eduardo Castro-Ospina [1,*], Miguel Angel Solarte-Sanchez [1], Laura Stella Vega-Escobar [1], Claudia Isaza [2] and Juan David Martínez-Vargas [3]

[1] Grupo de Investigación Máquinas Inteligentes y Reconocimiento de Patrones, Instituto Tecnológico Metropolitano, Medellín 050013, Colombia; miguelsolarte244621@correo.itm.edu.co (M.A.S.-S.); lauravega@itm.edu.co (L.S.V.-E.)
[2] SISTEMIC, Electronic Engineering Department, Universidad de Antioquia-UdeA, Medellín 050010, Colombia; victoria.isaza@udea.edu.co
[3] GIDITIC, Universidad EAFIT, Medellín 050022, Colombia; jdmartinev@eafit.edu.co
* Correspondence: andrescastro@itm.edu.co

Abstract: Sound classification plays a crucial role in enhancing the interpretation, analysis, and use of acoustic data, leading to a wide range of practical applications, of which environmental sound analysis is one of the most important. In this paper, we explore the representation of audio data as graphs in the context of sound classification. We propose a methodology that leverages pre-trained audio models to extract deep features from audio files, which are then employed as node information to build graphs. Subsequently, we train various graph neural networks (GNNs), specifically graph convolutional networks (GCNs), GraphSAGE, and graph attention networks (GATs), to solve multi-class audio classification problems. Our findings underscore the effectiveness of employing graphs to represent audio data. Moreover, they highlight the competitive performance of GNNs in sound classification endeavors, with the GAT model emerging as the top performer, achieving a mean accuracy of 83% in classifying environmental sounds and 91% in identifying the land cover of a site based on its audio recording. In conclusion, this study provides novel insights into the potential of graph representation learning techniques for analyzing audio data.

Keywords: ecoacoustics; environmental sound classification; graph neural networks; graph representation learning; node classification; pre-trained models

Citation: Castro-Ospina, A.E.; Solarte-Sanchez, M.A.; Vega-Escobar, L.S.; Isaza, C.; Martínez-Vargas, J.D. Graph-Based Audio Classification Using Pre-Trained Models and Graph Neural Networks. *Sensors* **2024**, *24*, 2106. https://doi.org/10.3390/s24072106

Academic Editor: Hector Eduardo Roman

Received: 21 February 2024
Revised: 22 March 2024
Accepted: 23 March 2024
Published: 26 March 2024

1. Introduction

Graphs are powerful mathematical structures that have been extensively employed to model and analyze complex relationships and interactions across various domains [1]. In passive acoustic monitoring applications, which help to create conservation plans, ecoacoustics has recently gained great importance as a cost-effective tool to analyze species conservation and ecosystem alteration. In this field, it is necessary to analyze a large amount of acoustic data to assess variations in the ecosystem. Moreover, in recent years, the field of graph representation learning has grown due to the increased interest in using these graph structures for learning and inference tasks [2]. To learn from graphs, it is crucial to develop algorithms and models that can efficiently capture and make use of the detailed structural information present in graph data. These approaches have found applications in diverse fields, including bioinformatics, computer vision, recommendation systems, and social network analysis [3–6].

Graph neural networks (GNNs) have emerged as a prominent class of models for learning on graphs, offering distinct advantages over traditional artificial intelligence techniques [7]. Unlike traditional methods that operate on independent data points, GNNs use the inherent connectivity and dependencies within the graph structure to learn and

propagate information across nodes. By recursively aggregating and transforming node features based on their local neighborhood, GNNs can capture both local and global patterns, enabling them to model complex relationships in graph data effectively. Notably, significant advancements in tasks such as node classification, link prediction, and graph generation have been made by leveraging their ability to capture structural dependencies [8].

Automatic audio classification tasks have attracted attention in recent years, specifically the classification of environmental sounds [9], enabling applications ranging from speech recognition [10,11] to soundscape ecology [12,13]. Traditional classification techniques such as *k*-nearest neighbors, support vector machines, and neural network classifiers have been used [14–17]. However, its performance mostly relies on hand-crafted features from representations as temporal, spectral, or spectro-temporal domains. Moreover, deep learning techniques using 1D (raw waveform) [18–21] or 2D (spectrograms) [22–25] convolutional neural networks (CNN) have shown significant improvements over hand-crafted methods. Nevertheless, these networks do not consider the relationships that may exist between different environmental sounds. Recurrent Neural Networks were initially proposed to capture feature dependencies from audio data [26–28]. More recently, Transformer models have emerged to model longer feature dependencies and leverage parallel processing [29–32]. Transformer models can handle variable input lengths and utilize attention mechanisms, making them aware of the global context and allowing their application on audio classification tasks.

Although graphs have been widely employed to represent and analyze visual and textual data, their potential to represent audio data has received relatively less attention [33–35]. Nonetheless, audio data, ranging from speech signals to music recordings, inherently exhibit temporal dependencies and complex patterns that can be effectively captured and modeled using graph-based representations. Working with graphs presents several challenges in their construction and subsequent processing. Determining how to generate feature information for each node and establishing connections between nodes in the network remain open problems. In this study, we propose utilizing pre-trained audio models to extract informative features from audio files, enabling the building of graphs that capture the inherent relationships and temporal dependencies present in the audio data.

Specifically, this study aims to address the problem of audio classification as a node classification task over graphs. To achieve this, we propose the following approach: (i) characterizing each audio with pre-trained networks to leverage transfer learning from models trained on large amounts of similar data, (ii) constructing graphs with each set of generated features, and (iii) utilizing the constructed graphs to classify nodes into predefined categories, taking advantage of their relationship. To accomplish this, we will use two datasets, a public one and one acquired in a passive acoustic monitoring study. We will evaluate the performance of three state-of-the-art GNNs: convolutional graph networks (GCN), graph attention networks (GAT), and GraphSAGE. These models leverage the rich structural information encoded in audio graphs in a transductive manner to learn discriminative representations capable of efficiently distinguishing between different audio classes. By comparing the performance of these models, we attempt to evaluate which of the graph models performs better on audio classification tasks.

In conclusion, this study contributes to the emerging field of graph representation learning by exploring the application of GNNs for audio classification. In particular, we demonstrate the effectiveness of pre-trained audio models to generate node information for graph representations and compare the performance of three GNN architectures. The results not only advance the state-of-the-art in audio classification but also emphasize the potential of graph-based approaches for modeling and analyzing complex audio data.

2. Graph Neural Networks

A graph is a widely used data structure, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consisting of nodes $\left(\mathcal{V} = \left\{v_1, v_2, \ldots, v_{|\mathcal{V}|}\right\}\right)$ and edges $(\mathcal{E} = \{e_{ij}\})$ representing a link between node $i\,(v_i)$ and node $j\,(v_j)$. A useful way to represent a graph is through an adjacency matrix

$\left(A \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}\right)$, where the presence of an edge is encoded as an entry with $A_{ij} = 1$ if there is an edge between (v_i) and (v_j) and $A_{ij} = 0$ otherwise. Additionally, each node i has associated feature information or embeddings denoted as $h_i^{(0)}$.

GNNs are machine learning methods that receive data in the form of graphs and use neural message passing to generate embeddings for graphs and subgraphs. In [2], the author provides an overview of neural message passing, which can be expressed as follows:

$$h_u^{(k+1)} = update^{(k)}\left(h_u^{(k)}, agg^{(k)}(\{h_v, \forall\, v \in N(u)\})\right). \quad (1)$$

In this equation, $h_u^{(k)}$ is the current embedding of node u where the embeddings (h_v) of neighboring nodes will be sent; $N(u)$, the neighborhood of node u; and $update^{(k)}$ and $agg^{(k)}$, permutation-invariant functions.

There exist various GNN models that differ in their approach to the *aggregation* or *update* function expressed in Equation (1) and in their ability to perform prediction tasks at node, edge, or network level [36]. The theory of the three GNN models used in this study is presented below.

2.1. Graph Convolutional Networks (GCNs)

The goal of GCNs is to generalize the convolution operation to graph data by aggregating both self-features and neighbors' features [37]. Following the update rule given by Equation (2), GCNs enforce self-connections by making $\tilde{A} = A + I$ and stack multiple convolutional layers followed by nonlinear activation functions.

$$H^{(k+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(k)}W^{(k)}\right) \quad (2)$$

In this equation, H is the feature matrix containing the embeddings of the nodes as rows, and $\tilde{D}$ denotes the degree matrix of the graph, which is computed as $\tilde{D}_{ij} = \sum_j \tilde{A}_{ij}$. Moreover, $\sigma(\cdot)$ is an activation function, and W is a trainable weight matrix.

2.2. Graph SAmple and aggreGatE (GraphSAGE)

GraphSAGE, a framework built on top of the original GCN model [38], updates each node's embedding information by sampling the number of neighbors at different hop values and aggregating their respective embedding information. This iterative process allows nodes to increasingly gain information from different parts of the graph.

The main difference between the GCN model and GraphSAGE lies in the aggregation function. Where GCNs use an average aggregator, GraphSAGE employs a generalized aggregation function. Also, in GraphSAGE, self-features are not aggregated at each layer. Instead, the aggregated neighborhood features are concatenated with self-features, as shown in Equation (3).

$$h_u^{(k+1)} = \sigma\left(\left[W^{(k)}agg\left(\{h_v^{(k)}, \forall\, v \in N(u)\}\right), B^{(k)}h_u^{(k)}\right]\right) \quad (3)$$

In this equation, B is a trainable weight matrix, and agg denotes a generalized aggregation function, such as mean, pooling, or LSTM.

2.3. Graph Attention Networks (GATs)

In GCNs (Equation (2)), graph node features are averaged at each layer, with weights determined by coefficients obtained from the degree matrix $(\tilde{D})$. This implies that the outcomes of GCNs are highly dependent on the graph structure. GATs [39], for their part, seek to reduce this dependency by implicitly calculating these coefficients, taking into account the importance assigned to each node's features using the attention mechanism [40]. The purpose of this is to increase the model's representational capacity.

The expression for GATs is presented in Equation (4).

$$h_u^{(k+1)} = \sigma\left(\sum_{v \in N(u)} \alpha_{uv} W^{(k)} h_v^{(k)}\right) \tag{4}$$

In this equation, α_{uv} represents the attention coefficients of the neighbors of node u, $v \in N(u)$, regarding the aggregation feature aggregation at this node. These coefficients are computed as

$$\alpha_{uv} = \frac{exp\left(a^\top LeakyReLU(W[h_u, h_v])\right)}{\sum_{j \in N(u)} exp\left(a^\top LeakyReLU\left(W\left[h_u, h_j\right]\right)\right)}, \tag{5}$$

with a denoting a trainable attention vector [41].

3. Materials

3.1. UrbanSound8K

UrbanSound8K is an audio dataset [42] that contains 8732 labeled audio files in WAV format and lasts four seconds or less. Each audio file belongs to one of the following ten classes: *air conditioner*, *car horn*, *children playing*, *dog bark*, *drilling*, *engine idling*, *gun shot*, *jackhammer*, *siren*, and *street music*.

The audio files are originally pre-distributed across ten folds, as depicted in Figure 1. To avoid errors that could invalidate the results and enable fair comparisons with existing literature, it is advised to perform cross-validation using the ten predefined folds.

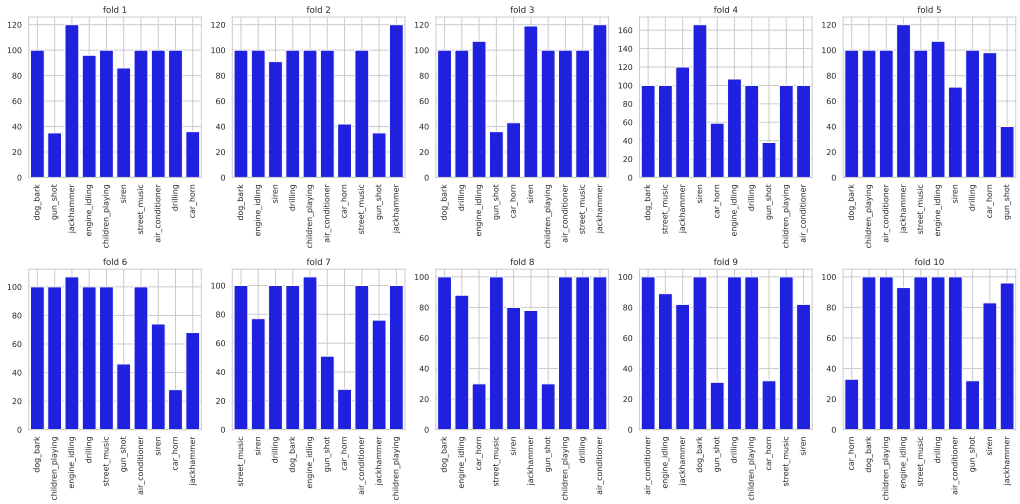

Figure 1. Distribution of the ten classes across the predefined folds.

3.2. Rey Zamuro Reserve

This dataset arises from a passive acoustic monitoring study conducted at Rey Zamuro and Matarredonda Private Reserves (3°31′02.5″ N, 73°23′43.8″ W), located in the municipality of San Martín in the Department of Meta, Colombia. The reserve covers an expanse of 6000 hectares, predominantly characterized by natural savanna constituting around 60%, interspersed with introduced pasture areas. The remaining 40% is covered by forests. This region falls within the tropical humid biome of the Meta foothills, showcasing an average temperature of 25.6 °C.

Data were acoustically recorded in September of 2022. A 13 × 8 grid was installed with 94 AudioMoth automatic acoustic devices placed 400 m from each other; of these recorders, one was not used due to deteriorated audio. The recording was made every fourteen minutes for seven consecutive days. The recordings were captured in mono format at a sampling rate of 192,000 Hz. The study encompassed various habitats, such

as forest interiors, edges, and adjacent areas, each with distinct characteristics, including undergrowth. The recording heights were standardized at 1.5 m above the ground.

Depending on the kind of land cover, each acoustic recording of Rey Zamuro soundscapes was classified as forest, savanna, or pasture. These labels were given based on the placement of each automated recording unit. A total of 71,497 recordings were obtained, of which 14,546 correspond to forest class, 14,994 to savanna class, and 41,957 to pasture class. In all, 80% of the dataset is used as the training set, and the remaining 20% as the test set.

3.3. Pre-Trained Models for Audio Feature Extraction

We use pre-trained deep learning audio models to extract deep features from each audio file, which will be used as node information in the constructed graphs, i.e., as the values $h_i^{(0)}$. Specifically, we employed the following three models: VGGish, YAMNet, and PANNs.

3.3.1. VGGish

VGGish is a pre-trained neural network architecture particularly designed to generate compact and informative representations, or deep embeddings, for audio signals [43]. It is inspired by the Visual Geometry Group (VGG) network architecture originally developed for image classification [44]. The deep embeddings generated by VGGish effectively capture relevant acoustic features and serve as a foundation for various audio processing tasks, such as audio classification, content-based retrieval, and acoustic scene understanding [45,46]. VGGish was trained on AudioSet [47], a publicly available and widely used large-scale audio dataset comprising millions of annotated audio clips and 527 classes, including animal sounds, musical instruments, human activities, environmental sounds, and more.

The architecture of VGGish consists of several layers, including convolutional, max-pooling, and fully connected layers. In this model, the processed audio is segmented into 0.96-second clips, and a log-Mel spectrogram is calculated for each clip, serving as the input to the neural network. Then, the convolutional layers apply a set of learnable filters to the input audio spectrogram, aiming to detect local patterns and extract low-level features. Following each convolutional layer, max-pooling layers are employed to reduce the spatial dimensions of the obtained feature maps while retaining the most important information. This process helps capture and preserve relevant patterns at different scales and further abstract the representations. Lastly, the final layers of VGGish, i.e., the fully connected layers, take the flattened output of the preceding convolutional and max-pooling layers and map it to a 128-dimensional representation. This mapping aims to capture global and high-level dependencies, resulting in deep embeddings that encode meaningful information about the audio signal and can serve as input for subsequent shallow or deep learning methods.

3.3.2. PANNs

Large-scale Pretrained Audio Neural Networks (PANNs) are pre-trained models specifically developed for audio pattern recognition [48]. Their architecture is built upon CNNs, which are well-suited for analyzing audio mel-spectrograms. PANNs have multiple layers, including convolutional, pooling, and fully connected layers. These layers work together to learn hierarchical representations of audio patterns at various levels of abstraction.

The training process of PANNs involves pre-training the model on the large-scale AudioSet dataset. By being trained on this dataset, PANNs learn to capture a wide range of audio patterns, making them strong audio feature extractors. These audio patterns are then mapped to a 2048-dimensional output space.

3.3.3. YAMNet

Yet another Audio Mobilenet Network (YAMNet) is a pre-trained neural network architecture that utilizes the power of deep CNNs and transfer learning to perform accurate and efficient audio analysis [49].

YAMNet is a mobilenet-based architecture consisting of a stack of convolutional layers, followed by global average pooling and a final fully connected layer with softmax activation. The convolutional layers extract local features by convolving small filters over the input audio spectrogram, thereby capturing different levels of temporal and spectral patterns. Then, the global average pooling operation condenses the extracted features into a fixed-length representation. Finally, the fully connected layer produces the classification probabilities for each sound class.

YAMNet's primary objective is to accurately classify audio signals into a wide range of sound categories. However, the embeddings obtained after the global average pooling operation can also be useful.

To process audio, YAMNet divides the audio into segments of 0.96 s with a hop length of 0.48 s. For each segment, a feature output comprising 1024 dimensions is generated.

3.4. Graph Construction

A popular way to determine the edges of a graph is to define whether two points are neighbors through the k-nearest neighbors (k-NN) algorithm. According to this method, the neighbors of node v_i are those k-nearest neighbors in the feature space [50]. Thus, the k-NN algorithm assigns edges between v_i and its neighbors.

4. Experimental Framework

The proposed methodology of this study to assess the effectiveness of using graphs to represent audio data by leveraging pre-trained audio models to generate node information is depicted in Figure 2, and involves the following stages: (i) VGGish, YAMNet, and PANNs pre-trained audio models are used to extract features from both datasets, (ii) those deep features are used independently to construct graphs where each node represents an audio file, and edges are determined based on the k-NN algorithm, and (iii) the constructed graphs are used to train and optimize certain hyperparameters on GCN, GraphSAGE, and GAT models to perform node classification.

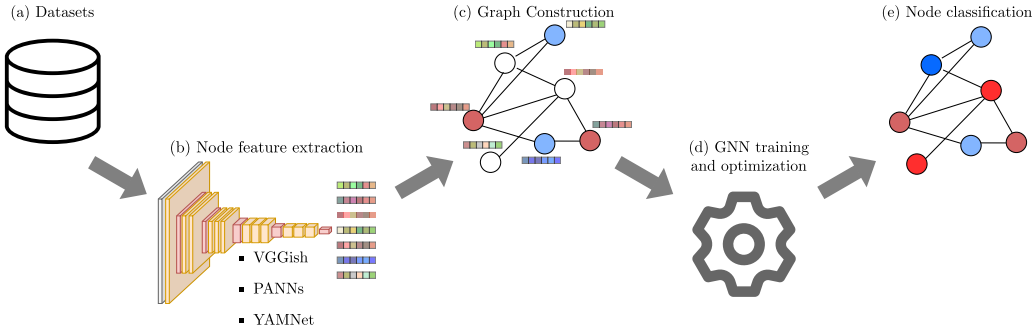

Figure 2. The workflow diagram proposed in this study illustrates that for each audio of a dataset (**a**) deep features are extracted with pre-trained audio models (**b**), then graphs are constructed by including those features as node information and setting edges with k-NN (**c**). For test data, the nodes present information but no labels (in the diagram the nodes unfilled are the test nodes). Subsequently, some GNN models are trained and optimized (**d**). Finally, trained models allow discriminating test nodes between classes (red or blue in the diagram) through transductive learning (**e**).

As a first step, we employed the VGGish, PANNs, and YAMNet pre-trained models to extract features from the audio files in both datasets to be used as node embedding vectors. In the UrbanSound8K dataset, fold information was preserved for the extracted features,

as shown in Figure 3. VGGish model generates a 128-dimensional deep feature vector for every 0.96 s of an audio clip, and YAMNet produces a 1024-dimensional deep feature vector for every 0.48 s. Since the audio files have a maximum duration of four seconds for UrbanSound8K and 60 s for Rey Zamuro, to obtain node embeddings of the same length, we averaged those 128-dimensional VGGish-based and 1024-dimensional YAMNet-based deep features.

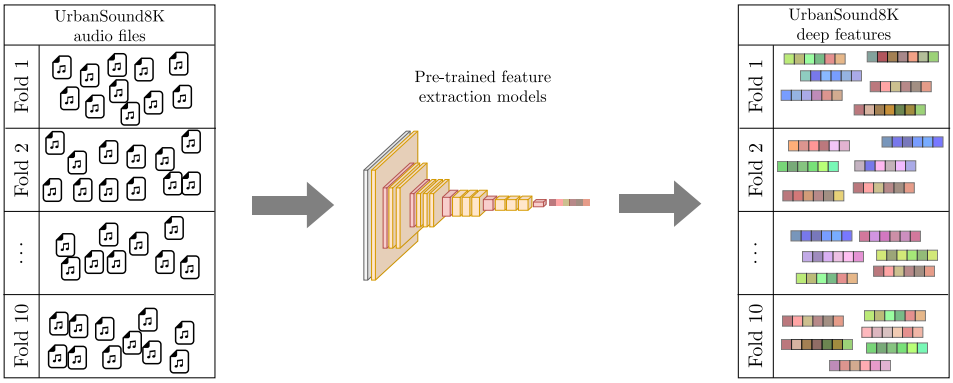

Figure 3. Feature extraction scheme. The audio files from each fold of the UrbanSound8K dataset were characterized using pre-trained models.

Subsequently, for each dataset characterized using the pre-trained models, we constructed a graph where the nodes represented the audio embeddings, and the edges were defined by applying the k-NN algorithm, where each node is connected with its k nearest neighbors. The value k was optimized for each architecture using Optuna [51]. Then, we implemented the GCN, GraphSAGE, and GAT architectures using PyTorch Geometric [52]. For the GCN and GraphSAGE models, we employed a two-layer architecture with a hidden dimension optimized by Optuna and an output dimension equal to the number of classes, i.e., three for the Rey Zamuro dataset and ten for UrbanSound8K. For the GAT model, we used a two-layer architecture, with the first layer having a value for hidden dimension optimized by Optuna and 10 heads, followed by a second layer with an output dimension corresponding to the number of classes and one head.

To compute the attention coefficients, we employed a slope of 0.2 on the LeakyReLU activation function in Equation (5). For all trained GNNs, we used the ReLU activation function and a dropout with a probability of 0.5. All models were trained to minimize cross-entropy loss using the Adam optimizer (with a learning rate of 0.001 and weight decay of 5×10^{-4}) for 300 and 1300 epochs for UrbanSound8K and Rey Zamuro dataset, respectively.

Finally, for UrbanSound8K, we evaluated the performance of the models in terms of accuracy using ten-fold cross-validation, i.e., following the dataset's distribution across the ten predefined folds. Alternatively, due to the large amount of data and the associated computational cost for training use, the performance of the models for the Rey Zamuro dataset was evaluated with the test set.

5. Results and Discussion

Tables 1 and 2 present the accuracy results of the three GNN models (GCN, GraphSAGE, and GAT) trained for audio file classification, with nodes representing the audio data in a graph. These nodes are characterized by three distinct feature sets derived from pre-trained models (VGGish, PANNs, and YAMNet) applied to UrbanSound8K and Rey Zamuro datasets. Additionally, the tables display the optimal hyperparameters determined by Optuna for each GNN model and node characterization combination. For the UrbanSound8K dataset, where fold distribution is predefined, accuracy results are presented as mean values accompanied by their corresponding standard deviations. Conversely, accuracy results for the Rey Zamuro dataset focus solely on the test set.

Table 1. UrbanSound8K accuracies.

Feature Model	GNN Architecture	Best Hyperparameters	Accuracy
VGGish	GCN	k = 10 n_hidden = 55	0.77 ± 0.04
	GraphSAGE	k = 12 n_hidden = 57	0.76 ± 0.05
	GAT	k = 9 n_hidden = 52	0.79 ± 0.05
YAMNet	GCN	k = 5 n_hidden = 196	0.81 ± 0.04
	GraphSAGE	k = 11 n_hidden = 55	0.8 ± 0.03
	GAT	k = 6 n_hidden = 252	0.82 ± 0.04
PANNs	GCN	k = 4 n_hidden = 40	0.83 ± 0.03
	GraphSAGE	k = 5 n_hidden = 183	0.82 ± 0.03
	GAT	k = 10 n_hidden = 206	0.83 ± 0.03

Table 2. Rey Zamuro accuracies.

Feature Model	GNN Architecture	Best Hyperparameters	Accuracy
VGGish	GCN	k = 5 n_hidden = 48	0.63
	GraphSAGE	k = 10 n_hidden = 63	0.63
	GAT	k = 6 n_hidden = 49	0.63
YAMNet	GCN	k = 5 n_hidden = 62	0.76
	GraphSAGE	k = 6 n_hidden = 56	0.74
	GAT	k = 6 n_hidden = 53	0.78
PANNs	GCN	k = 6 n_hidden = 64	0.87
	GraphSAGE	k = 7 n_hidden = 63	0.85
	GAT	k = 5 n_hidden = 51	0.91

The results reveal the consistent superiority of PANNs across both datasets and all three trained GNN models. In particular, on the Rey Zamuro dataset, PANNs show a significant improvement of up to 18% in accuracy. The higher performance can be attributed to the larger dimensional feature space produced by PANNs, with 2048 dimensions, compared to VGGish and YAMNet, which have dimensions of 128 and 1024, respectively. This larger feature space of PANNs is more suitable for capturing detailed information from audio data.

Furthermore, among the compared GNN models, GAT emerges as the top performer, demonstrating sustained superiority across both datasets. This underscores the effectiveness of the attention mechanism in exploiting graph information and optimizing aggregation strategies. Tables 3 and 4 present the computational costs of the experiments conducted, measured in terms of time and the number of trainable parameters of the networks for the UrbanSound8K and Rey Zamuro datasets, respectively. It is important to note that each model possesses a different number of neurons in the hidden layer due to the optimization performed with Optuna. The GAT model has the highest number of parameters for both datasets and the feature sets generated with the pre-trained models. Specifically, the largest GAT model for the UrbanSound8K dataset has 8M parameters when using PANNs' deep features. Regarding training time, the GAT model for this dataset can take up to 35 times longer than training GCN and GraphSAGE models. Concerning the Rey Zamuro dataset, we also calculate the time for each model under test. Once again, the GAT model demonstrates the largest number of parameters, as well as longer training and testing times. However, during testing, the times are closer to those of the other two models. Although training time can indeed be long, it is worth considering that a trained network can be scalable regardless of the amount of data. However, it is crucial to consider the computational requirements for building and storing the graph.

Our results show that representing audio datasets through graphs and using deep features extracted from pre-trained models as node features enables sound classification. However, it is important to acknowledge an ongoing research challenge in the graph-building step, particularly in setting its node feature information and edges. To the best of our knowledge, only one study has employed GNNs for sound classification on UrbanSound8K dataset [34]. In one such study, the overall classification accuracy obtained using GNNs was 63.5%, which improved to 73% when GNNs were used in combination with features learned from a CNN. However, our results surpass this, even in the case of GraphSAGE, whose lowest accuracy is 76% for VGGish features. Moreover, our findings are comparable to those reported in other studies employing 1D CNN models. For example, in [18], RawNet CNN was presented, which worked with the raw waveform and achieved an accuracy of 87.7 ± 0.2. Additionally, in [19], a CNN called EnvNet-v2 obtained an accuracy of 78.3%, in [20] with very deep 1D convolutional networks a maximum accuracy of 71.68% only for the 10th fold used as the test set, while in [21], a proposed end-to-end 1D CNN achieved 89% accuracy. In addition, 2D CNN models have also been used on the UrbanSound8K dataset, reaching 79% [22], 70% [23], 83.7% [24], and 97% [25]. It should be noted that although other studies used the UrbanSound8K dataset to train 1D or 2D CNNs, they often employ unofficial random splits of the dataset, conducting their own cross-validations or training-test splits. This causes them to use different training and validation data than published papers that follow the official distribution, making comparison unfair.

Table 3. Computational cost for UrbanSound8K dataset tests.

Feature Model	GNN Architecture	# Parameters	Training Time [s]
VGGish	GCN	7655	13.3
	GraphSAGE	15,799	10.9
	GAT	145,640	86.7
YAMNet	GCN	202,870	18.6
	GraphSAGE	113,805	33.0
	GAT	5,221,480	370.9
PANNs	GCN	82,370	12.7
	GraphSAGE	753,421	37.1
	GAT	8,487,240	453.9

Table 4. Computational cost for Rey Zamuro dataset tests.

Feature Model	GNN Architecture	# Parameters	Training Time [s]	Test Time [ms]
VGGish	GCN	6682	14.3	4.9
	GraphSAGE	17,461	25.8	12.9
	GAT	137,240	232.9	84.0
YAMNet	GCN	64,180	20.9	6.1
	GraphSAGE	115,874	68.4	80.4
	GAT	1,098,200	295.4	99.9
PANNs	GCN	131,786	27.2	7.8
	GraphSAGE	259,381	136.9	190.6
	GAT	2,101,240	325.7	120.3

6. Conclusions

In this paper, we explored using graphs as a suitable representation of acoustic data for sound classification tasks, focusing on the UrbanSound8K dataset and a passive acoustic monitoring study. Particularly, this study offers novel insights into the potential of graph representation learning methods for analyzing audio data.

First, we utilized pre-trained audio models, namely VGGish, PANNs, and YAMNet, to compute node embeddings and extract informative features. Then, we trained GCNs, GraphSAGE, and GATs and evaluated their performance. For the UrbanSound8K dataset, we employed a ten-fold cross-validation approach with the dataset's predefined folds for performance evaluation. Additionally, we partitioned the Rey Zamuro Dataset into train and test sets to validate its results. Moreover, during the training stage, we conducted hyperparameter optimization to attain the best possible model for the built graphs.

Our findings demonstrate the effectiveness of using graphs to represent audio data. In addition, they show that GNNs can achieve a competitive performance in sound classification tasks. Most notably, it is shown that it is possible to identify ecosystem states through audio and GNNs. Notably, the best results were obtained when employing PANNs-based deep features with the three GNN models. Among the GNN models, the GAT model outperforms the others. This advantage stems from its attention-based operation, enabling it to aggregate node information by assigning weights to its neighbors based on relevance.

To further our research, we plan to explore the feasibility of using temporal GNNs for sound classification tasks to leverage graphs constructed using deep features based on temporal segments of the audio signal, such as those obtained with VGGish and YAMNet. Additionally, the proposed methodology will be applied to the area of soundscape ecology, seeking to generate acoustic heterogeneity maps from the treatment of large volumes of data with GNN techniques that allow exploiting the acoustic relationships between different recording sites.

Author Contributions: Conceptualization, A.E.C.-O.; Formal analysis, A.E.C.-O., M.A.S.-S. and J.D.M.-V.; Methodology, A.E.C.-O., M.A.S.-S. and L.S.V.-E.; Project administration, C.I.; Supervision, J.D.M.-V.; Validation, L.S.V.-E. and C.I.; Writing—original draft, A.E.C.-O. and M.A.S.-S.; Writing—review and editing, L.S.V.-E., C.I. and J.D.M.-V. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Universidad de Antioquia, Instituto Tecnológico Metropolitano de Medellín, Alexander von Humboldt Institute for Research on Biological Resources, and Colombian National Fund for Science, Technology and Innovation, Francisco Jose de Caldas—MINCIENCIAS (Colombia). [Program No. 111585269779].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available under request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Veličković, P. Everything is connected: Graph neural networks. *Curr. Opin. Struct. Biol.* **2023**, *79*, 102538. [CrossRef] [PubMed]
2. Hamilton, W.L. Graph representation learning. In *Synthesis Lectures on Artifical Intelligence and Machine Learning*; Morgan and Claypool: San Rafael, CA, USA, 2020; Volume 14, pp. 1–159.
3. Angles, R.; Gutierrez, C. Survey of graph database models. *ACM Comput. Surv. (CSUR)* **2008**, *40*, 1–39. [CrossRef]
4. Goyal, P.; Ferrara, E. Graph embedding techniques, applications, and performance: A survey. *Knowl.-Based Syst.* **2018**, *151*, 78–94. [CrossRef]
5. Dong, G.; Tang, M.; Wang, Z.; Gao, J.; Guo, S.; Cai, L.; Gutierrez, R.; Campbel, B.; Barnes, L.E.; Boukhechba, M. Graph neural networks in IoT: A survey. *ACM Trans. Sens. Netw.* **2023**, *19*, 1–50. [CrossRef]
6. Su, X.; Xue, S.; Liu, F.; Wu, J.; Yang, J.; Zhou, C.; Hu, W.; Paris, C.; Nepal, S.; Jin, D.; et al. A comprehensive survey on community detection with deep learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**. [CrossRef] [PubMed]
7. Chen, F.; Wang, Y.C.; Wang, B.; Kuo, C.C.J. Graph representation learning: A survey. *APSIPA Trans. Signal Inf. Process.* **2020**, *9*, e15. [CrossRef]
8. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24. [CrossRef]
9. Bansal, A.; Garg, N.K. Environmental Sound Classification: A descriptive review of the literature. *Intell. Syst. Appl.* **2022**, *16*, 200115. [CrossRef]
10. Passricha, V.; Aggarwal, R.K. A hybrid of deep CNN and bidirectional LSTM for automatic speech recognition. *J. Intell. Syst.* **2019**, *29*, 1261–1274. [CrossRef]
11. Mustaqeem; Kwon, S. A CNN-assisted enhanced audio signal processing for speech emotion recognition. *Sensors* **2019**, *20*, 183. [CrossRef]
12. Dias, F.F.; Ponti, M.A.; Minghim, R. A classification and quantification approach to generate features in soundscape ecology using neural networks. *Neural Comput. Appl.* **2022**, *34*, 1923–1937. [CrossRef]
13. Quinn, C.A.; Burns, P.; Gill, G.; Baligar, S.; Snyder, R.L.; Salas, L.; Goetz, S.J.; Clark, M.L. Soundscape classification with convolutional neural networks reveals temporal and geographic patterns in ecoacoustic data. *Ecol. Indic.* **2022**, *138*, 108831. [CrossRef]
14. Kostrzewa, D.; Brzeski, R.; Kubanski, M. The classification of music by the genre using the KNN classifier. In *Beyond Databases, Architectures and Structures. Facing the Challenges of Data Proliferation and Growing Variety, Proceedings of the 14th International Conference, BDAS 2018, Held at the 24th IFIP World Computer Congress, WCC 2018, Poznan, Poland, 18–20 September 2018*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 233–242.
15. Prabavathy, S.; Rathikarani, V.; Dhanalakshmi, P. Classification of Musical Instruments using SVM and KNN. *Int. J. Innov. Technol. Explor. Eng.* **2020**, *9*, 1186–1190. [CrossRef]
16. Tsalera, E.; Papadakis, A.; Samarakou, M. Monitoring, profiling and classification of urban environmental noise using sound characteristics and the KNN algorithm. *Energy Rep.* **2020**, *6*, 223–230. [CrossRef]
17. Malik, H.; Bashir, U.; Ahmad, A. Multi-classification neural network model for detection of abnormal heartbeat audio signals. *Biomed. Eng. Adv.* **2022**, *4*, 100048. [CrossRef]
18. Li, S.; Yao, Y.; Hu, J.; Liu, G.; Yao, X.; Hu, J. An ensemble stacked convolutional neural network model for environmental event sound recognition. *Appl. Sci.* **2018**, *8*, 1152. [CrossRef]
19. Tokozume, Y.; Ushiku, Y.; Harada, T. Learning from between-class examples for deep sound recognition. *arXiv* **2017**, arXiv:1711.10282.
20. Dai, W.; Dai, C.; Qu, S.; Li, J.; Das, S. Very deep convolutional neural networks for raw waveforms. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 421–425.
21. Abdoli, S.; Cardinal, P.; Koerich, A.L. End-to-end environmental sound classification using a 1D convolutional neural network. *Expert Syst. Appl.* **2019**, *136*, 252–263. [CrossRef]
22. Salamon, J.; Bello, J.P. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Process. Lett.* **2017**, *24*, 279–283. [CrossRef]
23. Pons, J.; Serra, X. Randomly weighted cnns for (music) audio classification. In Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 336–340.
24. Zhang, Z.; Xu, S.; Cao, S.; Zhang, S. Deep convolutional neural network with mixup for environmental sound classification. In Proceedings of the Chinese Conference on Pattern Recognition and Computer Vision (PRCV), Guangzhou, China, 23–26 November 2018; pp. 356–367.
25. Su, Y.; Zhang, K.; Wang, J.; Madani, K. Environment sound classification using a two-stream CNN based on decision-level fusion. *Sensors* **2019**, *19*, 1733. [CrossRef]
26. Gong, Y.; Chung, Y.A.; Glass, J. Ast: Audio spectrogram transformer. *arXiv* **2021**, arXiv:2104.01778.

27. Gan, J. Music feature classification based on recurrent neural networks with channel attention mechanism. *Mob. Inf. Syst.* **2021**, *2021*, 7629994. [CrossRef]
28. Banuroopa, K.; Shanmuga Priyaa, D. MFCC based hybrid fingerprinting method for audio classification through LSTM. *Int. J. Nonlinear Anal. Appl.* **2021**, *12*, 2125–2136.
29. Zhuang, Y.; Chen, Y.; Zheng, J. Music genre classification with transformer classifier. In Proceedings of the 2020 4th International Conference on Digital Signal Processing, Chengdu, China, 19–21 June 2020; pp. 155–159.
30. Nogueira, A.F.R.; Oliveira, H.S.; Machado, J.J.; Tavares, J.M.R. Transformers for urban sound classification—A comprehensive performance evaluation. *Sensors* **2022**, *22*, 8874. [CrossRef]
31. Zhang, Y.; Li, B.; Fang, H.; Meng, Q. Spectrogram transformers for audio classification. In Proceedings of the 2022 IEEE International Conference on Imaging Systems and Techniques (IST), Kaohsiung, Taiwan, 21–23 June 2022; pp. 1–6.
32. Zhu, W.; Omar, M. Multiscale audio spectrogram transformer for efficient audio classification. In Proceedings of the ICASSP 2023—2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece, 4–10 June 2023; pp. 1–5.
33. Zhang, S.; Qin, Y.; Sun, K.; Lin, Y. Few-Shot Audio Classification with Attentional Graph Neural Networks. In Proceedings of the Interspeech 2019, Graz, Austria, 15–19 September 2019; pp. 3649–3653.
34. Aironi, C.; Cornell, S.; Principi, E.; Squartini, S. Graph-based representation of audio signals for sound event classification. In Proceedings of the 2021 29th European Signal Processing Conference (EUSIPCO), Dublin, Ireland, 23–27 August 2021; pp. 566–570.
35. Hou, Y.; Song, S.; Yu, C.; Wang, W.; Botteldooren, D. Audio event-relational graph representation learning for acoustic scene classification. *IEEE Signal Process. Lett.* **2023**, *30*, 1382–1386. [CrossRef]
36. Bishop, C.M.; Bishop, H. Graph Neural Networks. In *Deep Learning: Foundations and Concepts*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 407–427.
37. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
38. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11.
39. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
40. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
41. Brody, S.; Alon, U.; Yahav, E. How attentive are graph attention networks? *arXiv* **2021**, arXiv:2105.14491.
42. Salamon, J.; Jacoby, C.; Bello, J.P. A dataset and taxonomy for urban sound research. In Proceedings of the 22nd ACM international conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 1041–1044.
43. Hershey, S.; Chaudhuri, S.; Ellis, D.P.; Gemmeke, J.F.; Jansen, A.; Moore, R.C.; Plakal, M.; Platt, D.; Saurous, R.A.; Seybold, B.; et al. CNN architectures for large-scale audio classification. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 131–135.
44. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
45. Kim, B.; Pardo, B. Improving content-based audio retrieval by vocal imitation feedback. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 4100–4104.
46. Tsalera, E.; Papadakis, A.; Samarakou, M. Comparison of pre-trained cnns for audio classification using transfer learning. *J. Sens. Actuator Netw.* **2021**, *10*, 72. [CrossRef]
47. Gemmeke, J.F.; Ellis, D.P.; Freedman, D.; Jansen, A.; Lawrence, W.; Moore, R.C.; Plakal, M.; Ritter, M. Audio set: An ontology and human-labeled dataset for audio events. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 776–780.
48. Kong, Q.; Cao, Y.; Iqbal, T.; Wang, Y.; Wang, W.; Plumbley, M.D. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2020**, *28*, 2880–2894. [CrossRef]
49. Models/Research/Audioset/Yamnet at Master · Tensorflow/Models—github.com. Available online: https://github.com/tensorflow/models/tree/master/research/audioset/yamnet (accessed on 18 April 2023).
50. Maier, M.; Luxburg, U.; Hein, M. Influence of graph construction on graph-based clustering measures. *Adv. Neural Inf. Process. Syst.* **2008**, *21*, 1–8.
51. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2623–2631.
52. Fey, M.; Lenssen, J.E. Fast graph representation learning with PyTorch Geometric. *arXiv* **2019**, arXiv:1903.02428.