Semi-supervised graph learning for underwater source localization using ship-of-opportunity spectrograms

Jhon Castro-Correa, Mohsen Badiey, Jhony Giraldo, Fragkiskos Malliaros

Semi-Supervised Graph Learning for Underwater Source Localization Using Ship-of-Opportunity Spectrograms

Jhon A. Castro-Correa,[1, a] Mohsen Badiey,[1, b] Jhony H. Giraldo,[2, c] and Fragkiskos D. Malliaros[3, d]
[1] *Department of Electrical and Computer Engineering, University of Delaware, Newark, DE, USA.*
[2] *LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France*
[3] *Université Paris-Saclay, CentraleSupélec, Inria, Centre for Visual Computing (CVN), Gif-Sur-Yvette, France.*

(Dated: 3 February 2026)

Conventional techniques for underwater source localization have traditionally relied on optimization methods, matched-field processing, beamforming, and, more recently, deep learning. However, these methods often fall short to fully exploit the data correlation crucial for accurate source localization. This correlation can be effectively captured using graphs, which consider the spatial relationship among data points through edges. This work introduces a novel graph learning module for source localization using spectrograms from Ships-Of-Opportunity (SOO), which represent mid-frequency acoustic broadband signals from ship-radiated noise ranging from 360 to 1100 Hz, collected during the 2017 Seabed Characterization Experiment (SBCEX 2017). The proposed approach follows a two-step process: first, a pre-trained Convolutional Neural Network (CNN) module is used for feature extraction via self-supervised learning, and then a Graph Neural Network (GNN) model is trained using semi-supervised learning for source localization. The graph is constructed using a k-nearest neighbor (k-NN) algorithm, incorporating features extracted by the CNN from the spectrograms. By employing this two-stage training strategy, our framework addresses the challenge of limited labeled data availability while achieving performance comparable to conventional supervised learning models. The effectiveness of our approach is demonstrated through model evaluation on both synthetic and measured data, showcasing the architecture's ability to generalize well to unseen scenarios.

[https://doi.org/10.1121/10.0039042]

[Preprint] Pages: 1–14

I. INTRODUCTION

Accurate acoustic source localization presents a formidable challenge due to the abrupt fluctuations in the water column and the inherent acoustic noise in underwater environments[1]. In general, anisotropic signals are transmitted underwater to assess the propagation of acoustic broadband signals, infer underlying seabed properties, or locate static and moving sources using inverse problems[2,3]. However, among the signals of interest are those emitted by large container and oil tanker merchant ships, commonly referred to as Ships-of-Opportunity (SOO), which produce broadband noise as they pass near the experimental area. These large ships typically follow a constant speed and a straight track along shipping lanes relative to the Vertical Line Array (VLA). The acoustic signature pattern produced by SOO shows distinct parabola-like patterns in the time-frequency domain, describing the ship's trajectory moving away or towards the receiver array, with the parabola vertex representing the Closest Point of Approach (CPA) range (see Fig. 1 for further details). In previous works, the broadband noise from SOO has been the subject of study, leading to the development of a deterministic model for the broadband spectral shape by Wales and Heitmeyer[4]. Their model represents the noise resulting from the highly nonlinear interaction between turbulence from the ship hull and propeller motion over the frequency band of 30-1200 Hz. The propagation of broadband noise at a wide frequency band in the water column implies that the received SOO spectrograms contain substantial information about the seafloor[3], which can be harnessed to infer both seabed properties, ship speed, and the CPA range. Likewise, collected SOO signatures can be complemented by accurate information about the location and speed provided by the Automatic Identification System (AIS), enabling the estimation of source localization parameters and proving valuable for generating properly annotated data for machine learning frameworks[5,6].

[a] jcastro@udel.edu
[b] badiey@udel.edu
[c] jhony.giraldo@telecom-paris.fr
[d] fragkiskos.malliaros@centralesupelec.fr

The variability in the waveguide, including sound speed in the water column, water depth, bathymetry, and seabed properties, directly impacts tasks such as source localization, tracking, and seabed characterization. To address these problems, developed approaches must consider multiple scenarios in highly fluctuating environments. The crucial role of environmental properties in estimating the locations of moving acoustic sources aids in signal enhancement and underwater navigation. Traditional algorithms used in these tasks include beamforming[7,8], Matched Field Processing (MFP)[9–11], Monte Carlo sampling[12], simulated annealing[13], Gibbs sampling[14], or Bayesian Maximum Entropy (BME)[15], which incorporate prior knowledge about the waveguide. Recently, data-driven approaches have been explored for several applications in ocean acoustics via supervised learning[1,2,5,16] or unsupervised learning[17,18]. Several of the mentioned studies have applied deep-learning techniques for source localization using supervised, semi-supervised and unsupervised learning techniques[19–21], often focusing on a limited number of canonical sediments. For instance, Van Komen *et al.*[3,6] used CNNs to predict source localization parameters and classify four canonical seabed types using both explosive charges and SOO spectrograms. They demonstrated that supervised learning methods can determine source range and seabed type from a one-second pressure time series from explosive sources and broadband signals from SOO.

This work extends previous supervised learning approaches to a flexible semi-supervised setting that allows the use of a partially labeled dataset, consisting of SOO spectrograms from broadband signals recorded on VLAs. We evaluate the impact of environmental variability on our framework and estimate the moving source's location, first using simulated data and then with measured data collected from three different VLAs during the SBCEX 2017. Due to the lack of experimental data, the training dataset consists of generated spectrograms using normal mode theory[22]. The training data comprise spectrograms from synthetic broadband signals in the $360 - 1100$ Hz frequency band over a 20 minutes interval. In addition, the spectrograms are generated using six different seabed types, covering a set of environments found in the literature across multiple regions in the world, ranging from soft-muddy to hard-sandy sediments. For testing, we use synthetic sets, and 61 measured SOO spectrograms were collected during the SBCEX 2017 in the New England Mud Patch area. The Global Positioning System (GPS) data from the spotted ships in the experiment were retrieved via AIS[23], and they serve as labels during the training and testing stages. Even though some GPS data are available, these data are scarce, outdated, and typically not free for commercial ship containers. These types of scenarios create a need to develop data-driven methods that do not rely solely on fully labeled datasets. The proposed approach relaxes the constraints of supervised learning methods by relying only on partially labeled data, which agrees with real-world scenarios where GPS data are rarely available. Our framework incorporates a CNN architecture trained using synthetic data without any label information via self-supervised learning and a GNN model trained via semi-supervised learning with few labeled data, aiming to estimate the CPA range. The proposed approach is purely data-driven and focuses on (1) exploring self-supervised learning to extract relevant features from SOO spectrograms without the requirement of labeled samples, and (2) demonstrating the efficacy of GNNs in the context of acoustic source localization using ship-radiated noise features. We introduce fundamental concepts of self-supervised learning for feature extraction and graph signal processing and apply them to analyze ship-radiated noise signals. We showcase the suitability of GNNs for underwater source localization and show that the combination of self-supervised learning with semi-supervised learning on GNNs yields promising results in localizing acoustic sources using broadband SOO spectrograms. The main contributions of this work are:

- To our knowledge, we are the first to introduce graph neural networks for acoustic source localization (CPA range estimation) in underwater acoustics. By incorporating a learning component, our approach extends graph signal processing-based techniques previously used for source localization, resulting in improved performance to complex underwater scenarios[24].
- We implement a two-step framework using self-supervised and semi-supervised learning as an alternative to supervised learning for ocean acoustic applications. Our approach mitigates the existing problem of lack of labeled data commonly found in at-sea experiments.

The manuscript is organized into several sections. Section III provides detailed information about the acoustic source localization experiment and the measured data. Section IV elaborates on the simulated data used to train and evaluate the proposed framework. In section V, we introduce the fundamental concepts of Graph Signal Processing (GSP) and GNNs in the context of acoustic source localization. Section VI presents the proposed framework for localizing acoustic sources using GNNs. The results are summarized in section VII, and ablation studies in section VIII, followed by conclusions in Sec. IX.

II. RELATED WORK

Despite the significant advances in deep learning, many widely used approaches in ocean acoustics still heavily rely on fully labeled datasets to address source localization or seabed classification tasks[3,5]. In addition, obtaining adequate labeled data in at-sea experiments is often challenging, if not impossible, rendering supervised learning algorithms less effective for real-world problems unless synthetic data is utilized. Some work has been done to mitigate the lack of annotated data, including active[25,26], semi-supervised[27–29], and unsupervised

learning[30,31] methodologies. However, while these approaches address the problem of limited labeled data, they often compromise model performance compared to their supervised counterparts. One efficient solution to eliminate the need for labels during training is by using self-supervised learning for feature extraction combined with a task-specific network. In this machine learning technique, models learn from data without explicit labeling, leveraging the inherent data structure to generate implicit labels or supervisory signals[32]. It has been demonstrated that self-supervised learning models can achieve comparable results to supervised models and are powerful feature extractors, as evidenced in Refs. [33–36].

At the same time, the increasing collection of complex structural data has driven the adoption of novel graph-based methods for processing and analysis, leveraging the inherent relationships within structured data. GSP and GNNs have emerged as key techniques, especially in irregular domains like sensor networks[37–42]. These methodologies model graph data with signals living on the graph's nodes, capturing feature information alongside the data's underlying structure. GSP leverages graph structures to address underlying spatiotemporal relationships within data, utilizing information gathered at each node. Similarly, GNNs extend deep learning theories from regular grids to irregular domains and improve model performance by incorporating spatial relationships within data, showcasing their utility in tasks such as image and speech recognition[43], social network analysis[44], materials modeling[45], bioinformatics[46], and semi-supervised learning[47–51], demonstrating significant potential in domains such as ocean acoustics[18,24,52–57].

III. MERCHANT SHIPS DURING THE SBCEX 2017

This paper uses data collected by three VLAs that recorded anthropogenic signals and ship noise signatures during SBCEX 2017 in the New England Mud Patch region. SBCEX 2017 was an international, multidisciplinary, and multi-institutional research project aimed at comprehending underwater acoustic propagation, estimating seabed parameters, and observing horizontal variations in the acoustical field[58,59]. The VLAs were deployed in an area with a water depth of approximately 75 m, with slight variations from 72 to 78 m. Two VLAs, VLA1-MPL and VLA2-MPL, each equipped with 16 hydrophones, were deployed by the Marine Physical Laboratory of Scripps Institution of Oceanography (MPL), whereas VLA-UD, deployed by the University of Delaware, consisted of eight hydrophones. For this study, only data from the hydrophone located at a depth of 33 m above the seafloor and approximately at the center of the VLAs were used. The recording times and positions of each VLA are listed in Table I, whereas the location for each VLA is shown in Figure 1(a). VLA-UD was continuously recorded from March 07 to April 08, 2017 (approximately 32.1 days), while VLA1-MPL and VLA2-MPL had intermittent recording schedules, with a total recording time of approximately 9.1 and 7.5 days, respectively.

TABLE I. Information about the receivers.

Parameter	VLA1-MPL	VLA2-MPL	VLA UD
Latitude	N 40.470°	N 40.442°	N 40.460°
Longitude	W 70.597°	W 70.527°	W 70.563°
Starting date	Mar 22, 2017	Mar 23, 2017	Mar 07, 2017
Ending date	Apr 03, 2017	Apr 02, 2017	Apr 08, 2017
Recorded time	9.1 days	7.5 days	32.1 days

During the SCBEX 2017 experiment, 61 large container and oil tanker merchant ships were observed traversing in a straight trajectory along two distinct shipping lanes. Figure 1(a) illustrates the location of the experimental area, situated near the northern and southern shipping lanes on the east coast of the United States. Throughout the experiment, VLA1-MPL, VLA2-MPL, and VLA-UD recorded radiated noise from numerous merchant and container ships as they pass through the southern and northern shipping lanes. The ship motion along the shipping lanes produces intrinsic patterns presented in the time-frequency domain at the low and medium frequencies bands, as depicted in Fig. 1(b)-(d). The parabola shape of the spectrograms (see Fig. 1) arises from the motion of the ship traveling towards and away from the VLAs, where the center of the parabola corresponds to the CPA rang (r_{CPA}), which is the closest range between the ship and the VLA. To verify the correspondence of the ship signatures in the collected data, AIS data obtained from the Marine Cadastre were retrieved to be used to identify the merchant ships passing through the area[60]. MSC KALAMATA serves as an example of one of the ships recorded by all three VLAs. For this particular case, from AIS data, it was confirmed that the ship's speed remained constant at 16.7 kn, while r_{CPA} was calculated based on its location and the arrays' positions, as reported in Ref. [5]. The CPA for MSC KALAMATA was 5.90 km, 3.10 km, and 4.90 km for VLA1-MPL, VLA2-MPL, and VLA-UD, respectively. Figures 1 (b) to (d) show the spectrograms of MSC KALAMATA recorded at the three VLAs and different geotimes, displaying striations with vertices occurring at r_{CPA} (dashed white line in the spectrograms). These well-defined striations span a wide frequency band on the spectrogram, generated with a time spacing of five seconds and a frequency step of 6.1 Hz.

IV. DATA PROCESSING

A. Mathematical notation

In this paper, sets are denoted by calligraphic letters such as $\mathcal{V}$, with $|\mathcal{V}|$ representing their cardinality. Matrices are shown in bold uppercase letters like $\mathbf{X}$, while vectors are denoted by bold lowercase letters like $\mathbf{x}$. The identity matrix is $\mathbf{I}$, and $\mathbf{1}$ is a vector of ones with proper dimensions. Transposition is represented by $(\cdot)^{\mathsf{T}}$, and

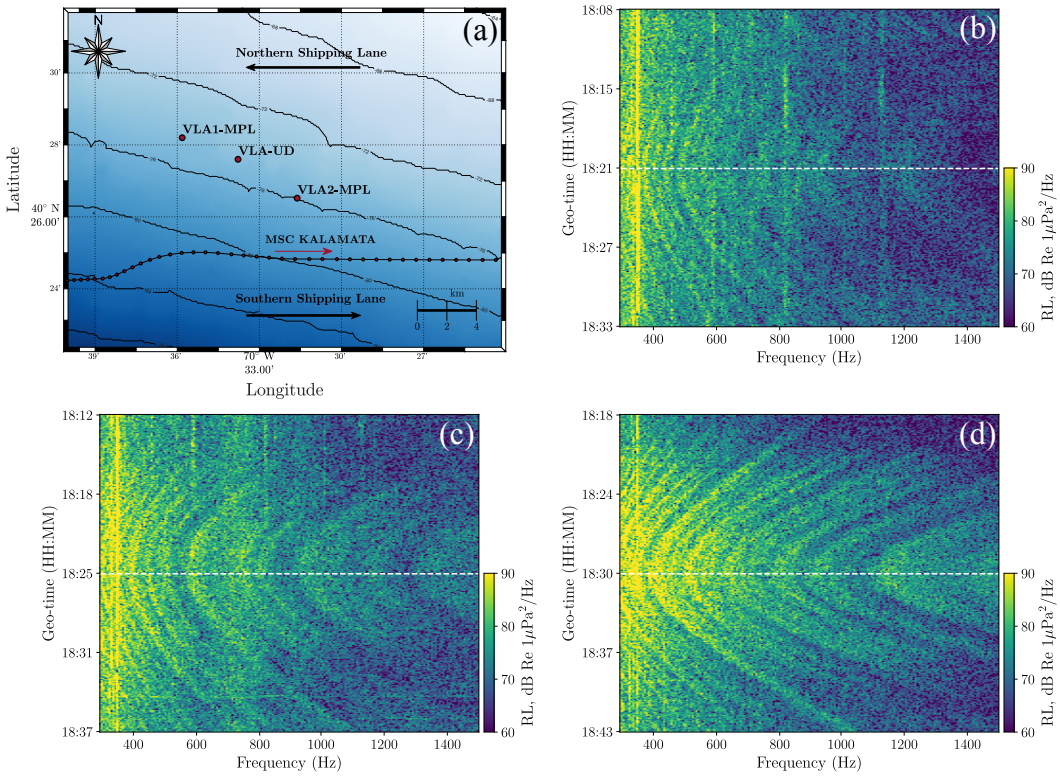

FIG. 1. Map of the experimental area during SBCEX 2017 showing position and SOO recording from MSC KALAMATA at different locations. (a) Track of MSC KALAMATA retrieved from AIS data and recorded signal on March 24, 2017, around 18:00 UTC at (b) VLA1-MPL, (c) VLA2-MPL, and (d) VLA-UD, for which the calibration factor is not 100% accurate. Spectrograms are extracted as a function of geo-time (UTC) and frequency for each VLA.

diag($\mathbf{x}$) is a diagonal matrix with $\{x_0, x_1, \ldots, x_{N-1}\}$ as its diagonal elements. Finally, the ℓ_2-norm of a vector is expressed as $\|\cdot\|_2$.

B. Synthetic data generation

This work evaluates the effectiveness of self-supervised learning and semi-supervised GNNs in estimating the CPA range of an acoustic moving source relative to a VLA. While these approaches require fewer labeled data than conventional supervised learning methods, they still require a sufficient quantity of diverse samples for effective training. To ensure variability on the data, we use the ORCA normal modes model to generate synthetic SOO spectrograms[61]. The spectral shape of broadband noise generated by transiting ships is accurately captured by the Wales and Heitmeyer equation[4], which is expressed as:

$$s(f) = s_0 - 10\log(f^{3.594}) + 10\log((1 + (f/340)^2)^{0.917}), \quad (1)$$

where s_0 represents the baseline source level of the ship in dB. However, it is important to note that the exact spectral levels of s_0 vary based on each ship's specific characteristics, and adds an extra parameter for simulation. This issue is addressed by normalizing the dataset, as later detailed in Section IV C.

The synthetic dataset serves as training data for the machine learning models introduced in Section V. To match the experimental setup, for data simulation, the receiver is placed 33 m above the seafloor, inside a water column with a depth of 75 m. These synthetic data include types of seabed sediment layers to reflect measured data collected around the world. The selection of seabed types was based on a measure of acoustic similarity, and the chosen categories represent six different sediment types ranging from mud to sand (refer to Table III in Ref. [5] for further details).

To account for environmental variability in the water column, the training dataset encompasses cases combining the six seabed types with ten different SSPs across the water column, measured during SBCEX 2017. The sound speed profiles have a range from 1470.5 m/s to 1472.4 m/s, as depicted in Figure 4 Ref. [5]. This combination of resulted in 60 distinct environments.

By incorporating diverse environments and varying source-receiver placements in data generation, we aim to capture the full range of values observed in the measured data samples. To cover the parameter space completely, certain labels for CPA range and ship speed are

fixed across all environments. Additional values for CPA range, ship speed and source depth are chosen randomly from a uniform distribution within the parameter ranges outlined in Table II. These combination, resulted in 405 different source parameters for each environment, leading to 24,300 data samples used to train the deep learning models proposed in this work. We use generated unlabeled data to train a feature extractor architecture via self-supervised learning and then utilize r_{CPA} to guide the training of the GNN model in a semi-supervised manner, leveraging only a small percentage of labels from the training data.

TABLE II. Source parameter selection for the synthetic dataset. Some fixed and random values for the CPA range and ship speed are used for each environment. All combinations of these three source parameters produce 405 data samples for each environment.

	Bounds	Fixed parameters	Random parameters
r_{CPA} (km)	[0.5 - 15]	5	10
v_{sh} (kn)	[8 - 20]	3	6
z_s (m)	[6 - 12]	0	3

C. Noise addition and data normalization

To address the issue of source level variations in measured data and to mitigate the impact of unknown s_0 values for the spotted merchant ships (see Table II in Ref. [5]), a two-step data preprocessing is employed during training involving frequency-dependent noise and data normalization. Considering that background noise varies with frequency, we use the realistic wind noise spectrum introduced by Hildebrand[62] (refer to noise level #2 in Fig. 1 of Ref. [62] for further details).

For each epoch, when a data sample is drawn, a different random noise signal is added. Each random noise signal is drawn from a uniform distribution $U \sim (a, b)$ with $a = 0$ dB and b equal to the maximum value in the wind noise spectrum, conforming a matrix that matches the dimensions of the input spectrograms. This additive noise matrix is then applied to each spectrogram $\mathbf{X} \in \mathbb{R}^{M\times N}$ when it is drawn during training, as $\mathbf{X} = \mathbf{X} + \mathbf{N}_{\text{rand}}$.

After the noise addition, the data is normalized based on the standard deviation of each sample. The spectrogram is initially converted from decibels (dB) to linear units (μPa) by removing the logarithmic scale. The standard deviation (σ) for each spectrogram $\mathbf{X} \in \mathbb{R}^{M\times N}$ with M time samples and N frequencies is calculated as:

$$\sigma_{\text{spec}} = \left(\frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(\mathbf{X}(t_i, f_j) - \bar{\mathbf{X}} \right)^2 \right)^{1/2} \tag{2}$$

The spectral density at time t_i and frequency f_j is expressed as $\mathbf{X}(t_i, f_j)$ in the spectrogram, with units of Re 1 μPa^2/Hz.

Finally, the normalized spectral densities in dB units for a spectrogram $\mathbf{X}_l$ are calculated as:

$$\mathbf{X}_l = 20 \log \left(\frac{\mathbf{X}}{\sigma_{\text{spec}}} \right). \tag{3}$$

When completing the preprocessing steps, notice that the unknown s_0 term has been discarded, alleviating the need for an extra parameter during data generation. This results in synthetic spectrograms that capture the realistic features, without losing any relevant feature. The visual representation in Fig. 2 showcases a spectrogram obtained after applying the described preprocessing steps.

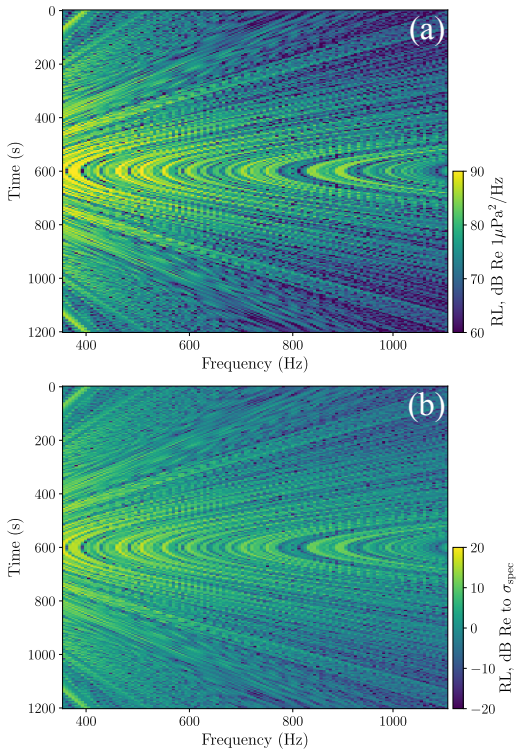

FIG. 2. Examples of spectrograms before and after preprocessing. (a) Spectrogram without any processing, (b) the same spectrogram after preprocessing. Initially, the spectrum is transformed to Pascals. Following this, an additive uniform noise level is introduced to the spectrogram. Subsequently, the spectrogram is transformed back to the dB scale, and the resulting values are scaled by the standard deviation of the spectrogram, denoted as σ_{spec}. This ensures that the values are normalized, preventing the model from being affected by large variations in the input data.

V. METHODOLOGY

As briefly mentioned in previous sections, the proposed framework in this paper considers a two-step approach, including (I) self-supervised learning for feature extraction and (II) semi-supervised learning on graph neural networks to estimate the CPA range. The following subsections will describe in detail both steps for the methodology.

A. Feature extraction via self-supervised learning

Self-supervised learning (SSL) is a paradigm where models are trained to learn meaningful representations from unlabeled data. This is achieved by designing tasks that the model can learn from without explicit supervision. One prominent approach in SSL is similarity contrastive learning (SimCLR), which leverages a contrastive loss function to encourage the model to learn representations that capture semantic similarities between data points[36]. SimCLR learns robust and discriminative representations by maximizing the similarity between augmented views of the same sample while minimizing the similarity between views of different images, as shown in Fig. 3.

The cornerstone of SimCLR is its contrastive loss function, which encourages the model to project similar views of a sample close to each other in the embedding space while pushing dissimilar views apart. Let $\mathbf{x}_i$ and $\mathbf{x}_j$ be two data instances, and let $\mathbf{z}_i = g(\mathbf{x}_i)$ and $\mathbf{z}_j = g(\mathbf{x}_j)$ be their respective representations learned by an encoder function $g(\cdot)$. The SimCLR cost function aims to maximize the agreement between the representations of similar data points while penalizing those that are dissimilar, as is formulated as:

$$\mathcal{L}(\mathbf{x}_i, \mathbf{x}_j) = -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / T)}{\sum_{k \neq i} \exp(\mathbf{z}_i \cdot \mathbf{z}_k / T)}, \tag{4}$$

where T is a temperature parameter that controls the influence of similarity scores. Intuitively, the loss encourages a high dot product between similar representations or "positive pairs" (e.g., different augmentations of the same samples) while penalizing the dot product between dissimilar ones or "negative pairs" (*e.g.*, samples from different classes). By minimizing this contrastive loss, the model learns representations that encode meaningful semantic relationships within the data, translating to improved performance in downstream tasks. SimCLR utilizes data augmentation techniques to generate both positive and negative pairs.

It is worth noting that in this paper, spectrograms are used as input samples instead of images, and the employed data augmentation techniques are specifically tailored for spectrogram data, as detailed in Ref. [1]. The augmentation strategies include composite time stretching, uniform noise, time warping, and time and frequency masking, each with different probabilities.

In this work, the time stretching augmentation involves changing the speed or duration of an audio signal without affecting its pitch. A batch is selected with probability p, and the samples are stretched by a constant factor d before being cropped to fit the network. To maintain the original size without using padding, d is set to $d \geq 1$.

Likewise, the ambient noise augmentation simulates potential noise-related artifacts in in-situ measurements. A batch is probabilistically selected, and random noise, drawn from a uniform distribution $\mathcal{N}$, is applied to each spectrogram during every epoch. The noise level for each spectrogram is determined by $N_{\text{noise}} = c \times H$ dB, where a random matrix $\eta \sim \mathcal{N}(\mu, \sigma)$ is scaled with a factor c (in dB).

Finally, the time warping, time masking, frequency masking, and combined transformations are adapted from Park *et al.*[63]. These techniques, originally designed for modifying spectrograms in speech recognition tasks, are applied within the context of ocean acoustics. To apply time warping, a segment of the spectrograms is selectively stretched in time while maintaining a constant frequency[63]. For each sample, a time warp parameter τ is defined, and a segment with a maximum width τ is randomly chosen from the spectrogram for stretching. The time-warping transformation enhances the network's robustness against deformations in the time direction induced by acoustic channel degradation. On the other hand, time masks, represented as small blocks with a maximum width w_t, result in the dropout of values within the mask. The parameter w_t is randomly selected from the interval $[0, \zeta]$, where ζ is a positive integer. Up to n_t masks are applied across the x-axis to each spectrogram. Similarly, up to n_f frequency masks with a height of h_f are applied along the y-axis. Time and frequency masking augmentations are important because they prepare the network for the partial loss of small speech segments and frequency information, which is particularly relevant to data collected from at-sea experiments. The final transformation combines time warping with time and frequency masking. Each spectrogram in the training set undergoes modification through time warping, followed by the addition of masking blocks across consecutive frequency channels and the time direction. The combined transformation that combine time warping, frequency masking, and time masking, is employed for good results, as demonstrated in Ref. [63].

B. Graph signal processing

1. Graph Signal Representation

A graph can be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{0, 1, \ldots, N-1\}$ is the set of nodes such that $|\mathcal{V}| = N$, and $\mathcal{E} = \{(i, j)\}$ the set of edges. The connectivity of the graph is represented by the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where a $\mathbf{A}_{(i,j)} > 0 \; \forall \; (i, j) \in \mathcal{E}$ indicates the weight of the edge between nodes i and j, and $\mathbf{A}_{(i,j)} = 0$, otherwise. In this work, we consider $\mathcal{G}$ to be undirected and weighted. $\mathbf{D} = \text{diag}(\mathbf{W1})$ is the degree matrix of $\mathcal{G}$, where each element $\mathbf{D}(i, i)$ of the diagonal is the sum of all edge weights incident at the ith node. Moreover, a graph signal is a function that maps a set of nodes to real values, $x : \mathcal{V} \to \mathbb{R}$, thus a one-dimensional[64] graph signal can be represented by $\mathbf{x} \in \mathbb{R}^N$. In GSP, the definition of a proper graph shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$ is a fundamental step for the downstream tasks. Typical choices of graph shift operators are the adjacency matrix $\mathbf{A}$ and the positive semi-definite combinatorial Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ matrix, along with their normalized forms.

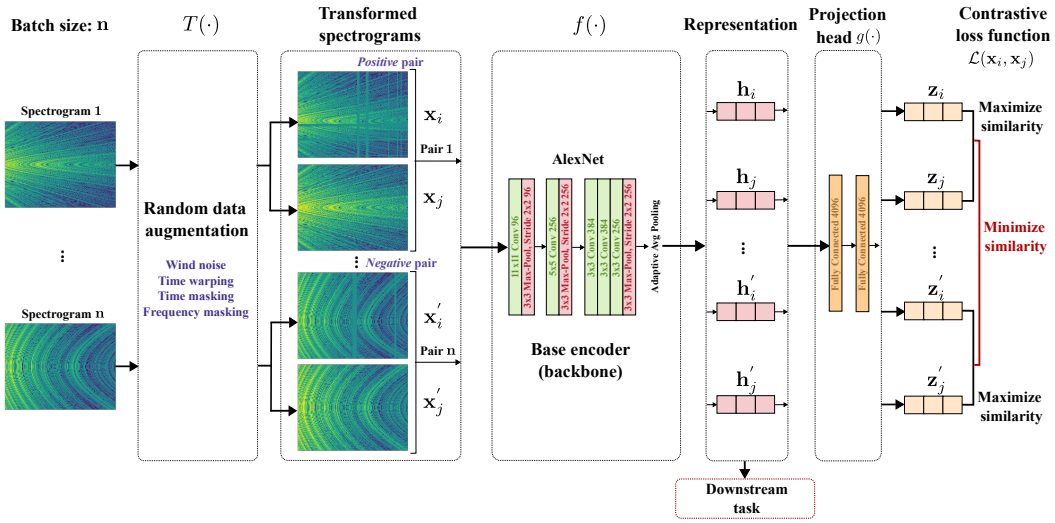

FIG. 3. During training, every spectrogram in a mini-batch passes through the first three blocks: the data augmentation module, the base encoder, and the projection head. This process generates two representation vectors for each original spectrogram (features), which are then used as input for the projection head to calculate the error.

2. Graph Convolutional Filters

A graph convolutional filter is defined as the linear combination of powers of the shift operators and the input graph signal[65]. Given a set of parameters $\mathbf{h} = [h_0, h_1, \ldots, h_K]$, a graph filter of order K is a linear mapping $\mathcal{H} : \mathbf{R}^N \to \mathbf{R}^N$ such that:

$$\mathcal{H}(\mathbf{x}) = \sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x}, \tag{5}$$

where $\mathbf{h}$ is the coefficients of the polynomial filtering function.

In essence, a GNN can be conceptualized as a composition of mapping functions. Each layer of the network consists of a graph convolutional filter, represented by Eq. (5), followed by an activation function $\sigma(\cdot)$. In practical implementations, the graph convolutional filter is typically multiplied by a matrix of learnable weights $\mathbf{W}$ and a bias term (see equation 6 for reference), following the approach introduced by Defferrard *et al.* for GNNs in Ref. [47].

C. Graph Neural Networks

In this study, we leverage the Chebyshev convolutional operator (ChebyConv)[47] as our propagation rule. The ChebyConv operator is based on continuous B-Chebyshev basis functions, which enables faster computation time and independence from kernel size. This operator aggregates node features $\mathbf{x}_i$ within local neighborhoods weighted by a trainable continuous kernel function. As the node features represent features on an irregular geometric structure, their spatial relations are locally defined by pseudo-spatial coordinates. Hence, when aggregating feature values in a node's neighborhood, the content of the complex space is utilized to determine how the features are aggregated, and the content of $\mathbf{x}_i$ defines what is aggregated. The ChebyConv propagation rule can be formulated as follows:

$$\mathbf{H}^{(l+1)} = \sigma\left(\sum_{k=0}^{K-1} \theta_k^{(l)} T_k^{(l)}(\tilde{\mathbf{L}}) \mathbf{H}^{(l)} \mathbf{W}^{(l)}\right), \tag{6}$$

where $\tilde{\mathbf{L}}$ is the scaled Laplacian $\tilde{\mathbf{L}} = 2\mathbf{L}/\lambda_{\max} - \mathbf{I}_n$, $\theta_k^{(l)}$ is the vector with Chebyshev coefficients, $T_k^{(l)}(\tilde{\mathbf{L}})$ is the Chebyshev basis, $\mathbf{H}^{(l)}$ is the matrix of activations such that $\mathbf{H}^{(0)} = \mathbf{X} \in \mathbb{R}^{N \times M}$ (input matrix of M-dimensional features), $\mathbf{W}^{(l)}$ is the matrix of trainable weights in layer l, and $\sigma(\cdot)$ is an activation function such as ReLU or softmax. A pictorial representation of the GNN employed in this work is shown in Fig. 4.

D. Semi-Supervised Learning for Graph Neural Networks

Semi-supervised learning emerges as a powerful paradigm for GNNs, strategically incorporating both labeled and unlabeled data to enhance model performance. GNNs, with their ability to capture relationships within graph-structured data, are particularly well-suited for semi-supervised tasks.

GNNs offer distinct advantages over CNNs, ResNets, and other conventional convolutional-based models for semi-supervised learning, particularly in contexts involving irregular or non-Euclidean data. Unlike CNN-based models that require grid-structured inputs and extensive labels, GNNs can naturally incorporate the relationships between labeled and unlabeled nodes through message passing. This allows them to effectively generalize from limited labeled data by leveraging the underlying graph

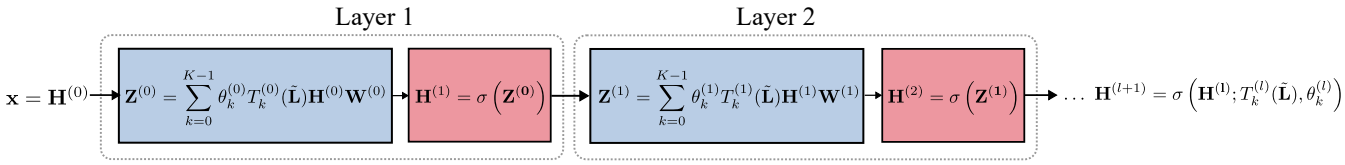

FIG. 4. A graph neural network is a composite of graph convolutional filters followed by activation functions.

structure. Additionally, GNNs are more flexible in handling variable input sizes and topologies, making them especially suited for applications like environmental sensing or physical system modeling. The objective is to predict labels for unlabeled nodes by leveraging the available partial labels. Formally, let $\mathbf{X} \in \mathbb{R}^{N\times M}$ denote the node feature matrix, where N is the number of nodes, and M is the number of features per node. Additionally, $\mathbf{y} \in \mathbb{R}^{N\times 1}$ represents the label vector for the regression task. Before training, three masks are created for training, validation, and testing. The objective is to omit labels for those nodes during training while retaining the inter-feature relationships provided by the features and edges of the graph. Training is performed using the labels from $\mathbf{Y}_{\text{train}} = \mathbf{J} \circ \mathbf{Y}$, where $\mathbf{J} \in \{0, 1\}$ is a random binary mask with density m and $\circ$ is the Hadamard product. For validation, we use the node embeddings $\mathbf{X}$ and a subset of labels selected from the training set, denoted as $\mathbf{Y}_{\text{val}} \subset \mathbf{Y}_{\text{train}}$. For testing, we use the complement of the training set labels, $\mathbf{Y}_{\text{test}} = \mathbf{Y}_{\text{train}}^{\complement}$. The validation set is used to compute errors during training, and the test set is employed to assess the model's generalization.

GNNs operate on graph-structured data where nodes represent samples and edges encode pairwise similarity. This enables the model to propagate label information through the graph via message passing, making GNNs particularly effective in semi-supervised settings with limited labeled data. In our framework, this allows the model to leverage both labeled and unlabeled spectrograms by exploiting inter-feature relationships, enhancing generalization and robustness in complex underwater environments.

VI. PROPOSED FRAMEWORK

Our approach uses a graph-based model to solve the downstream task of source localization. The GNN in charge of the localization task works on top of a graph constructed out of extracted features that contain the overall information of all the spectrograms in the dataset. The pipeline of our algorithm is shown in Fig. 5 and can be summarized in five steps: (1) Data preprocessing, (2) self-supervised pre-training and feature extraction using convolutional neural networks, (3) graph construction, (4) semi-supervised learning using a GNN, and (5) node-level regression task for CPA range estimation.

A. Feature extraction via Convolutional Neural Networks

Feature extraction is performed over 20-minute spectrograms generated from the simulated ocean response obtained from the ORCA normal mode model. The spectrograms are computed using a 5 s time interval window with a frequency spacing of 6.1 Hz. To ensure consistency and enhance pertinent features, preprocessing is applied, involving signal filtering within the $360 - 1100$ Hz frequency band. This process yields a matrix with 243×123 spectral density levels with time on the x-axis and frequency on the y-axis (refer to Ref. [5] for further details). Additionally, each spectrogram is normalized based on its standard deviation, drawing from prior work[2] and following steps in Section. IV.

The generated spectrograms are used as input for the pretraining stage of an AlexNet architecture. This architecture comprises five convolutional layers combined with max-pooling layers and fully connected layers to solve the downstream task (see Fig. 4 for details of the architecture). Before being used as a feature extractor, AlexNet undergoes a pretraining stage via self-supervised learning, as outlined in Section V. In a self-supervised setting, AlexNet learns meaningful features and predicts aspects of the input data without relying on external labels by leveraging a contrastive loss, as shown in Fig. 3.

After the pretraining stage, feature extraction is accomplished by removing the final dense layers. The remaining layers' output serves as a high-level representation or embedding from spectrograms and encapsulates relevant information for the downstream task. In the presented framework, we leverage the pre-trained AlexNet to generate the feature vectors to construct a graph that aims to capture intrinsic relationships in the spectrograms from the dataset.

To avoid overfitting and task-specific bias, we removed the fully connected layers from the AlexNet architecture after self-supervised pretraining (refer to Appendix A in the supplementary material). The convolutional layers retain high-level, spatially meaningful representations suitable for downstream generalization, while the dense layers—trained for specific classification tasks—may impose unnecessary constraints on the learned embeddings. Additionally, eliminating dense layers reduces the dimensionality of the feature vectors and the number of trainable parameters in the GNN, which is especially beneficial in semi-supervised settings with limited labeled data[36,66].

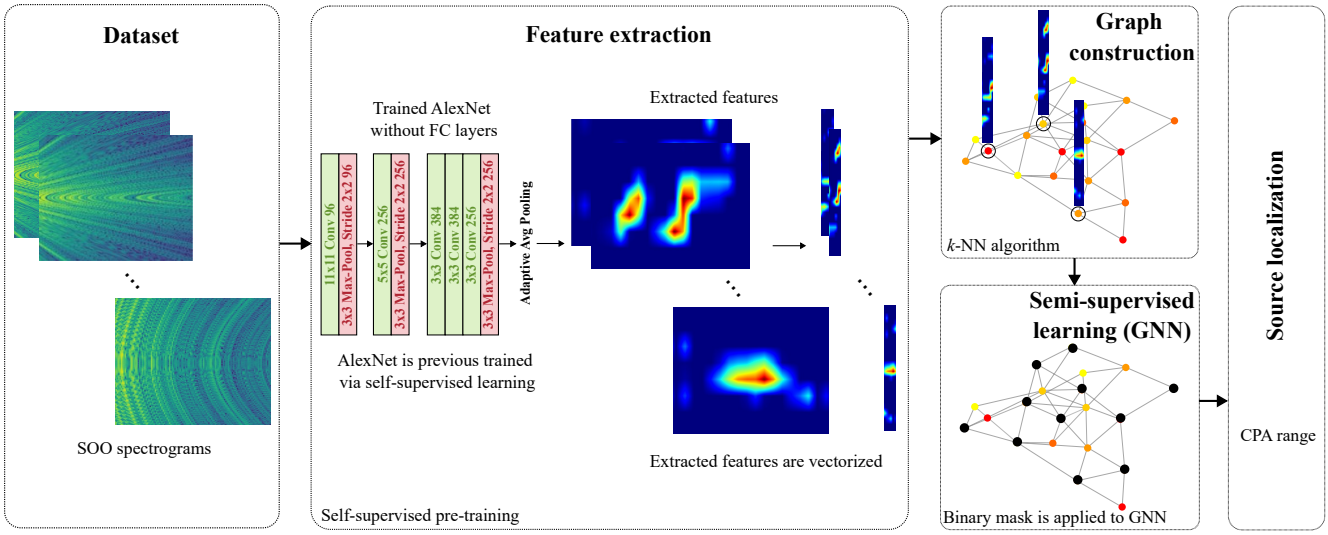

FIG. 5. The pipeline of our framework begins with the preprocessing of spectrograms. Subsequently, a feature extractor architecture, trained using self-supervised learning, is employed to extract meaningful information from each sample. These extracted features serve as the foundation for constructing a graph, which is then trained using semi-supervised learning. The primary objective of this training is to estimate the CPA range between the acoustic source and the receiver.

B. Feature-based graph construction

The features extracted by the pre-trained AlexNet architecture are then used to construct a weighted graph $G = (\mathcal{V}, \mathcal{E})$ with N nodes, where a node corresponds to a spectrogram in the dataset. Each node is connected to the closest k nodes using a k-nearest Neighbor (k-NN) algorithm based on the Euclidean distance between the M-dimensional feature vectors extracted by AlexNet. Let $\mathbf{F} \in \mathbb{R}^{N\times M}$ be the matrix of features for all samples in the dataset, such that $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_N]^T$, where $\mathbf{f}_i \in \mathbb{R}^M$ is the vector with features of the ith node. The weights for each edge (i, j) in the graph are defined using the Gaussian function:

$$\mathbf{W} = \exp\left(-\frac{\|\mathbf{f}_i - \mathbf{f}_j\|_2^2}{\sigma^2}\right), \tag{7}$$

where σ is the standard deviation computed as $\sigma = \frac{1}{|\mathcal{E}|}\sum_{(i,j)\in\mathcal{E}} \|\mathbf{f}_i - \mathbf{f}_j\|_2$, and $\|\mathbf{f}_i - \mathbf{f}_j\|_2^2$ is the Euclidean distance between feature vectors of nodes i and j.

C. Graph Neural Network architecture

The large graph constructed using the extracted features from AlexNet is used to solve the source localization task. The graph is processed using a GNN that builds upon the Chebyshev graph operator introduced in Ref. [47], and as outlined in Sec. V. The GNN architecture is comprised of l_{conv} convolutional layers, each with n_{units} hidden units. Each layer is powered by a ηth order Chebyshev polynomial filter and we use dropout with a percentage p to avoid overfitting during training. The architecture is trained using semi-supervised learning to solve the regression task at the node level.

VII. RESULTS

In this section, we report the performance of our proposed framework on a graph constructed from both synthetic and measured data at sea. Specifically, we present the results obtained after training, validating, and testing our model.

A. Evaluation metrics

In this paper, we employ diverse metrics to compare and assess the performance of the algorithms presented. For a ground truth vector $\mathbf{x}$ with N CPA ranges and their estimated values $\tilde{\mathbf{x}}$, we compute the Root Mean Square Error (RMSE), which quantifies the overall error magnitude as $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{x}}_i - \mathbf{x}_i)^2}$. Simultaneously, we calculate the Mean Absolute Error (MAE), a measure that accounts for error magnitude without considering direction, expressed as $\frac{1}{N}\sum_{i=1}^{N}|\tilde{\mathbf{x}}_i - \mathbf{x}_i|$. Furthermore, we utilize the Mean Absolute Percentage Error (MAPE) to evaluate errors with the magnitude of estimated values. It is computed as $\frac{1}{N}\sum_{i=1}^{N}\frac{|\tilde{\mathbf{x}}_i - \mathbf{x}_i|}{\tilde{\mathbf{x}}_i}$.

B. Implementation details and hyperparameter tunning

We implemented our framework using the PyTorch and PyTorch Geometric libraries[67], leveraging a GPU Nvidia Quadro RTX 5000 with 16 GB RAM for training the feature extractor and GNN. During the pre-training process, we employed the Cosine Annealing learning rate scheduler to optimize the training steps adaptively. To optimize the performance of the GNN used in our experiments, we conducted extensive hyperparameter tun-

ing by exploring a parameter space defined by several key parameters. For the number of convolutional layers l_{conv}, we varied the range from $\{1, \ldots, 5\}$, allowing us to investigate the impact of layer depth on the model's performance. The order η for the Chebyshev polynomial filters was changed between $\{1, \ldots, 5\}$. The number of hidden units n_{hidden} for graph convolutions was explored in the $\{2, \ldots, 20\}$ to assess the influence of feature dimensions. Regarding the learning rate ξ, we selected values in $[0.001, 0.1]$ to optimize the models' convergence during training. To mitigate overfitting and enhance generalization, we varied the dropout rate p within the range of $[0.01, 0.5]$. The final hyperparameters were selected as follows: $l_{\text{conv}} = 2$, $\eta = 3$, $n_{\text{hidden}} = 12$, $\xi = 0.01$, and $p = 0.05$, with a plateau learning rate schedule.

C. Evaluation in SOO data

In the evaluation of GNNs on synthetic data, our findings showcase promising results when comparing performance with state-of-the-art methods. Specifically, we compare our framework with a five-layer vanilla CNN model as shown in previous works[2,3,5]. As described in previous sections, the synthetic dataset used for training encompasses diverse scenarios and challenges, allowing for a comprehensive assessment of the GNN's capabilities.

The entire SOO dataset was used to train AlexNet via self-supervised learning. In this setting, the similarity between data samples is used to train the model; thus, no label information is required during the process. Features extracted using the pre-trained network were then used to construct a graph that leverages the inter-feature relationships in the data. The GNN performs node regression using Chebyshev polynomial filters to estimate the CPA range via semi-supervised learning. For this stage, train, validation, and test masks were created to use only partial information from the graph at each step, as described in Sec. V. In total, 30% of the nodes were used for training and validation, whereas the remaining 70% of the graph was used for testing. From the training nodes, we employed only 10% of labels during training and relied on the features and graph structure. In this way, the graph provides useful information from the dataset without the need for a complete set of labeled data, mitigating the problem of a lack of annotated data.

We tested our algorithm in unseen synthetic data during training and measured data from the 61 merchant ships spotted during the experiment. The results for CPA range estimation for both synthetic and measured SOO spectrograms are shown in Fig. 6. The green dots correspond to the predictions for each of the samples in the dataset, and the dashed red line represents the expected value. From the results, we observe that the GNN effectively estimates the CPA range from all the input spectrograms and leverages its ability to model complex relationships and dependencies within graph-structured data. Quantitative results are also presented in Table III. The RMSE, MAE, and MAPE are reported for our framework using 10% and 100% of labels from the training set to tune the GNN. Similarly, we are reporting the performance of the CNN, ResNet-18 and VGG-16 using 10% and 100% of labeled SOO spectrograms, trained in a supervised fashion. From the results, it is noticeable that the pre-trained AlexNet + GNN model outperforms the other models when using a smaller subset of labels. This suggests that the GNN excels in capturing intricate patterns and information embedded in synthetic datasets when few labeled data is available, showcasing its adaptability and robustness.

TABLE III. Performance comparison of different architectures for CPA range regression. Values in parentheses represent the percentage of labels used during training.

Model	Labels Used	RMSE	MAE	MAPE
Vanilla CNN	10%	0.652	0.413	0.101
	100%	0.358	0.235	0.074
ResNet-18	10%	0.667	0.419	0.107
	100%	0.362	0.242	0.081
VGG-16	10%	0.714	0.452	0.125
	100%	0.424	0.312	0.098
Ours (CNN+GNN)	10%	**0.474**	**0.305**	**0.096**
	100%	**0.371**	**0.241**	**0.089**

The results indicate that our approach outperforms existing fully supervised methods in terms of CPA range estimation when only few labeled information is available, highlighting the potential of our architecture for a variety of applications in ocean acoustics. Additionally we compare our approach with conventional architectures in a semi-supervised learning setup using pseudo-labeling[68] (refer to Appendix B in the supplementary material). The GNN's success in handling synthetic data highlights its potential applicability to real-world scenarios where understanding intricate relationships and dependencies is crucial. Similarly, the robust performance of the GNN on measured, real-world data underscores its capability to bridge the divide between synthetic simulations and actual operational environments, making it a promising choice for applications requiring the adept handling of complex, real-world data.

Our results highlight a key limitation of purely convolutional models in this task. While architectures such as ResNet-18 and VGG-16 effectively extract hierarchical features, they treat each input spectrogram independently and cannot leverage contextual relationships across samples. In contrast, the GNN component in our framework models inter-sample similarity via the graph structure, allowing the propagation of label information and enhancing generalization under sparse supervision. This ability is particularly valuable in real-world underwater acoustic settings, where labeled data (e.g., from AIS) is limited. Our findings suggest that augmenting CNN-based models with graph-based reasoning, as done in our method, is a more effective approach for semi-supervised regression tasks involving complex environmental data.

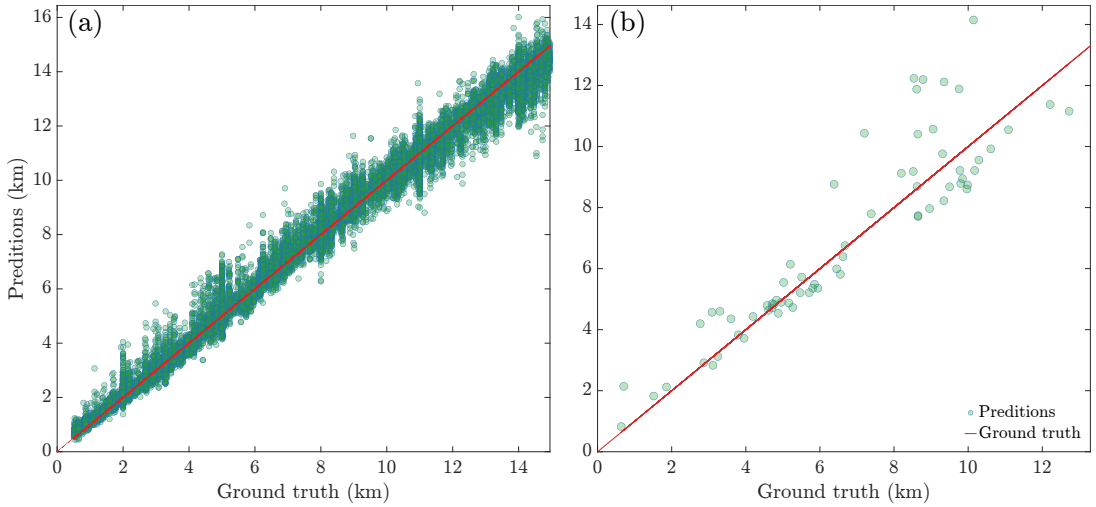

FIG. 6. Performance of the proposed framework in (a) synthetic dataset, and (b), in measured data. A solid red line denotes the ground truth values where predictions are shown as green points in the plots.

VIII. ABLATION STUDIES

To evaluate the performance of our architecture, we conducted several ablation studies. Specifically, we designed four ablation studies to assess (1) the graph construction strategy, (2) the performance of the GNN in comparison to a multi-layer perceptron architecture as a regressor, (3) the performance of GNN concerning the use percentage of labels, and (4) regarding time and frequency parameters for spectrogram generation used during training. For the latest ablation, refer to Appendix C in the supplementary material.

A. k-NN vs. learned graph from data

We conducted an ablation study focusing on the graph construction aspect, specifically comparing the performance of a k-NN approach against a data-driven method that learns the graph structure assuming features to the nodes to be smooth signals[69]. Let $\mathbf{X} \in \mathbb{R}^{N\times M}$ be the feature matrix consisting of N instances, where each instance is represented by an M-dimensional vector, and $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^\mathsf{T}$. Each $\mathbf{x}_i \in \mathbb{R}^M$ is the feature vector of the ith instance. Alternatively, $\mathbf{X}$ can be viewed as M one-dimensional graph signals, with $\mathbf{X} = [\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_M]$, and $\mathbf{x}'_i \in \mathbb{R}^N$ representing the ith column of $\mathbf{X}$. Then, the graph is learned as follows:

$$\arg\min_{\mathbf{A}\in\mathcal{A}} \|\mathbf{A} \circ \theta_k \mathbf{Z}\|_{1,1} - \mathbf{1}^\mathsf{T} \log(\mathbf{A}\mathbf{1}) + \frac{1}{2}\|\mathbf{A}\|_F^2, \qquad (8)$$

where $\mathcal{A}$ is the set of valid adjacency matrices (symmetric, with a zero diagonal, and non-negative), $\mathbf{Z} \in \mathbb{R}+^{N\times N}$ is the pairwise distances matrix of $\mathbf{X}$ such that $\mathbf{Z}(i,j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2^2; \forall 1 \leq i,j \leq N$, and θ_k is a parameter controlling the average number of neighbors of each node. For further details, please refer to Refs. [69, 70].

For the k-NN graph, we used $k = 10$ edges for each node, whereas for the learned graph, we fine-tuned the method to match, on average, $k = 10$ edges per node. The training for both graphs was done using 10% of the labels from the training set. The RMSE obtained during inference on the test set for each model is presented in Table IV.

TABLE IV. Ablation study for graph construction. The values in parentheses represent the percentage of labels used from the training set.

Architecture	RMSE
k-NN graph (10%)	0.474
learned graph (10%)	0.532

These results indicate that the k-NN graph outperformed the data-driven alternative yielding better performance.

B. Graph Neural Networks versus multi-layer perceptron for regression

In this case, we modified the final stage of our framework by replacing the ChebyNet GNN architecture with a conventional multi-layer perceptron (MLP). We used a two-layer MLP to perform the regression task for CPA range estimation and compared it with the optimal GNN architecture presented in the results section. We tried the GNN architecture using semi-supervised learning following the setting described in Section VII with 10% and 100% of labels, whereas the MLP was trained using supervised learning with 10% and 100% of the available training data. RMSE from both models during inference in test data is presented in Table V.

The performance metrics presented in the table clearly show the potential of GNN for performing the downstream task, outperforming the conventional MLP in the low-labeled data regime. The rationale behind

TABLE V. Ablation study for regressor performance. The values in parenthesis represent the percentage of labels used from the training set.

Architecture	RMSE
MLP (10%)	0.592
MLP (100%)	0.368
GNN (10%)	0.474
GNN (100%)	0.371

this behavior is that GNN can provide more information than MLPs due to the integration of weighted edges that capture feature relationships among nodes in the graph.

C. Performance versus percentage of number of labels

In this ablation study, different percentages of labels (i.e., masked nodes) were considered during the training of the GNN to assess the optimal number of labels for training. Table VI presents the computed RMSE for various percentages of labeled nodes from the dataset used during the training of the GNN model.

TABLE VI. Ablation study for the percentage of masked nodes. The percentages represent the number of labels used from the training set.

Percentage	RMSE	Percentage	RMSE	Percentage	RMSE
1%	0.821	**10%**	**0.474**	60%	0.416
2%	0.746	15%	0.467	70%	0.411
3%	0.712	20%	0.452	80%	0.401
4%	0.654	30%	0.431	90%	0.384
5%	0.612	40%	0.420	100%	0.371

By varying the percentage of covered nodes, it becomes evident that in the semi-supervised setting, reducing the percentage of labels does not significantly impact our model.

IX. CONCLUSION

We introduced GNNs for underwater ocean acoustic applications that involve leveraging graphs to estimate the CPA range between the source and receiver, facilitating the comprehensive representation of SOO spectrograms by capturing spatial and temporal relationships on the dataset.

We developed a framework that combines a two-step process, using a CNN as a feature extractor and a GNN to solve the downstream task of source localization for an acoustic moving source. Our input data consists of SOO spectrograms, and the CNN was trained in a self-supervised setting, eliminating the need for labeled data. In contrast, the GNN architecture underwent semi-supervised learning, allowing the utilization of a small percentage of labels during training. This approach not only addressed the challenges posed by limited labeled data in real-world scenarios but also enhanced the framework's ability to autonomously learn meaningful representations.

Through exhaustive experiments, we observed that our framework achieves performance on par with supervised algorithms such as vanilla CNNs, ResNet-18 and VGG-16. This underscores the effectiveness of our method in attaining comparable results to models trained with labeled data, showcasing the potential of self-supervised learning in capturing intricate patterns and semi-supervised learning in addressing data scarcity challenges. Ablation studies conducted in this work empirically demonstrated that GNNs outperform conventional models like MLPs in solving the downstream task.

It's worth noting that our algorithm requires training the CNN and GNN models separately. Consideration can be given to adopting end-to-end approaches to reduce training time burdens. Additionally, the GNN implemented in this work operates in a transductive setting.

ACKNOWLEDGMENTS

This work was partially supported by the Office of Naval grants (N00014-21-1-2760), (N00014-25-1-2074), (N00014-19-1-2539), by the ANR French National Research Agency under the JCJC projects GraphIA (ANR-20-CE23-0009-01), and DeSNAP (ANR-24-CE23-1895-01).

AUTHOR DECLARATION

Conflict of Interest

The authors have no conflict of interest to report.

DATA AVAILABILITY

Data supporting the findings of this study are available from the corresponding author upon reasonable request.

COPYRIGHT NOTICE

[1]J. A. Castro-Correa, M. Badiey, T. B. Neilsen, D. P. Knobles, and W. S. Hodgkiss, "Impact of data augmentation on supervised learning for a moving mid-frequency source," The Journal of the Acoustical Society of America **150**(5), 3914–3928 (2021) https://doi.org/10.1121/10.0007284 doi: 10.1121/10.0007284.

[2]T. B. Neilsen, C. D. Escobar-Amado, M. C. Acree, W. S. Hodgkiss, D. F. Van Komen, D. P. Knobles, M. Badiey, and J. Castro-Correa, "Learning location and seabed type from a moving mid-frequency source," The Journal of the Acoustical Society of America **149**(1), 692–705 (2021) doi: 10.1121/10.0003361.

[3]D. F. Van Komen, T. B. Neilsen, D. B. Mortenson, M. C. Acree, D. P. Knobles, M. Badiey, and W. S. Hodgkiss, "Seabed type and source parameters predictions using ship spectrograms in

convolutional neural networks," The Journal of the Acoustical Society of America **149**(2), 1198–1210 (2021) doi: 10.1121/10.0003502.

[4]S. C. Wales and R. M. Heitmeyer, "An ensemble source spectra model for merchant ship-radiated noise," The Journal of the Acoustical Society of America **111**(3), 1211–1231 (2002) doi: 10.1121/1.1427355.

[5]C. D. Escobar-Amado, T. B. Neilsen, J. A. Castro-Correa, D. F. Van Komen, M. Badiey, D. P. Knobles, and W. S. Hodgkiss, "Seabed classification from merchant ship-radiated noise using a physics-based ensemble of deep learning algorithms," The Journal of the Acoustical Society of America **150**(2), 1434–1447 (2021) https://doi.org/10.1121/10.0005936 doi: 10.1121/10.0005936.

[6]D. F. Van Komen, T. B. Neilsen, K. Howarth, D. P. Knobles, and P. H. Dahl, "Seabed and range estimation of impulsive time series using a convolutional neural network," The Journal of the Acoustical Society of America **147**(5), EL403–EL408 (2020) doi: 10.1121/10.0001216.

[7]J. C. Chen, K. Yao, and R. E. Hudson, "Source localization and beamforming," IEEE Signal Processing Magazine **19**(2), 30–39 (2002).

[8]C. Zhang, D. Florêncio, D. E. Ba, and Z. Zhang, "Maximum likelihood sound source localization and beamforming for directional microphone arrays in distributed meetings," IEEE Transactions on Multimedia **10**(3), 538–548 (2008).

[9]Y. Liu, R. Hu, D. Dou, H. Niu, D. Chen, and L. Xu, "Gaussian processes with normal-mode-based kernels for matched field processing," Applied Acoustics **220**, 109954 (2024).

[10]F. Hunter Akins and W. Kuperman, "Experimental demonstration of low signal-to-noise ratio matched field processing with a geoacoustic model extracted from noise," The Journal of the Acoustical Society of America **153**(5), 2818–2818 (2023).

[11]M. Asghari, M. Zareinejad, S. M. Rezaei, and H. Amindavar, "Robust matched field processing using an empirical characteristic function approach under impulsive noise environments," Acoustics Australia **51**(2), 201–219 (2023).

[12]M. Badiey, L. Wan, C. D. Escobar-Amado, and J. A. Castro-Correa, "Extraction of seabed properties in mid-frequency range using waveguide invariant applied to broadband signals," Proceedings of Meetings on Acoustics **45**(1), 070003 (2022) https://doi.org/10.1121/2.0001532 doi: 10.1121/2.0001532.

[13]O. T. Kosmas and D. Vlachos, "Simulated annealing for optimal ship routing," Computers & Operations Research **39**(3), 576–581 (2012).

[14]Z.-H. Michalopoulou and M. Picarelli, "Gibbs sampling for time-delay-and amplitude estimation in underwater acoustics," The Journal of the Acoustical Society of America **117**(2), 799–808 (2005) https://doi.org/10.1121/1.1847894 doi: 10.1121/1.1847894.

[15]D. P. Knobles, C. D. Escobar-Amado, M. J. Buckingham, W. S. Hodgkiss, P. S. Wilson, T. B. Neilsen, J. Yang, and M. Badiey, "Statistical inference of sound speed and attenuation dispersion of a fine-grained marine sediment," IEEE Journal of Oceanic Engineering **47**(3), 553–564 (2022) doi: 10.1109/JOE.2021.3091846.

[16]M. J. Bianco and P. Gerstoft, "Semi-supervised source localization with residual physical learning," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2022), pp. 3608–3612, doi: 10.1109/ICASSP43922.2022.9746564.

[17]M. Bianco and P. Gerstoft, "Dictionary learning of sound speed profiles," The Journal of the Acoustical Society of America **141**(3), 1749–1758 (2017) https://doi.org/10.1121/1.4977926 doi: 10.1121/1.4977926.

[18]J. A. Castro-Correa, S. A. Arnett, T. B. Neilsen, L. Wan, and M. Badiey, "Supervised classification of sound speed profiles via dictionary learning," Journal of Atmospheric and Oceanic Technology **40**(1), 99 – 112 (2023) https://journals.ametsoc.org/view/journals/atot/40/1/JTECH-D-21-0090.1.xml doi: 10.1175/JTECH-D-21-0090.1.

[19]H. Wen, C. Yang, D. Dou, L. Xu, and Y. Jiao, "Underwater source ranging by siamese network aided semi-supervised learning," JASA Express Letters **3**(9), 094803 (2023) https://doi.org/10.1121/10.0020991 doi: 10.1121/10.0020991.

[20]Y. Liu, H. Niu, Z. Li, and D. Zhai, "Unsupervised domain adaptation for source localization using ships of opportunity with a deep vertical line array," IEEE Journal of Oceanic Engineering **49**(1), 180–196 (2024) doi: 10.1109/JOE.2023.3301089.

[21]R. Long, J. Zhou, N. Liang, Y. Yang, and H. Shen, "Deep unsupervised adversarial domain adaptation for underwater source range estimation," The Journal of the Acoustical Society of America **154**(5), 3125–3144 (2023) https://doi.org/10.1121/10.0022380 doi: 10.1121/10.0022380.

[22]E. K. Westwood, C. T. Tindle, and N. R. Chapman, "A normal mode model for acousto-elastic ocean environments," The Journal of the Acoustical Society of America **100**(6), 3631–3645 (1996) doi: 10.1121/1.417226.

[23]AIS.

[24]P. Xiao and J. Yang, "A graph-based narrowband matched-field source localization method," JASA Express Letters **1**(12), 124803 (2021) https://doi.org/10.1121/10.0009060 doi: 10.1121/10.0009060.

[25]M. Prince, "Does active learning work? a review of the research," Journal of engineering education **93**(3), 223–231 (2004).

[26]L. Gao, W. Liu, K. Liu, and J. Wu, "Augsteal: Advancing model steal with data augmentation in active learning frameworks," IEEE Transactions on Information Forensics and Security (2024).

[27]A. Nzeyimana, "Improving kinyarwanda speech recognition via semi-supervised learning," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE (2024), pp. 10701–10705.

[28]Y. Long, Y. Li, S. Wei, Q. Zhang, and C. Yang, "Large-scale semi-supervised training in deep learning acoustic model for asr," IEEE Access **7**, 133615–133627 (2019).

[29]M. Alfarraj and G. AlRegib, "Semi-supervised learning for acoustic impedance inversion," in *SEG Technical Program Expanded Abstracts 2019* (Society of Exploration Geophysicists, 2019), pp. 2298–2302.

[30]H. Malik and H. Mahmood, "Acoustic environment identification using unsupervised learning," Security Informatics **3**, 1–17 (2014).

[31]W. Wang, R. Arora, K. Livescu, and J. A. Bilmes, "Unsupervised learning of acoustic features via deep canonical correlation analysis," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE (2015), pp. 4590–4594.

[32]A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," Technologies **9**(1), 2 (2020).

[33]X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," IEEE transactions on knowledge and data engineering **35**(1), 857–876 (2021).

[34]Y. Liu, M. Jin, S. Pan, C. Zhou, Y. Zheng, F. Xia, and S. Y. Philip, "Graph self-supervised learning: A survey," IEEE transactions on knowledge and data engineering **35**(6), 5879–5900 (2022).

[35]Y. Wang, C. M. Albrecht, N. A. A. Braham, L. Mou, and X. X. Zhu, "Self-supervised learning in remote sensing: A review," IEEE Geoscience and Remote Sensing Magazine **10**(4), 213–247 (2022).

[36]T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, PMLR (2020), pp. 1597–1607.

[37]I. Jabłoński, "Graph signal processing in applications to sensor networks, smart grids, and smart cities," IEEE Sensors Journal **17**(23), 7659–7666 (2017) doi: 10.1109/JSEN.2017.2733767.

[38] A. Ortega *et al.*, "Graph signal processing: Overview, challenges, and applications," Proceedings of the IEEE **106**(5), 808–828 (2018).

[39] X. Dong, D. Thanou, L. Toni, M. Bronstein, and P. Frossard, "Graph signal processing for machine learning: A review and new perspectives," IEEE Signal processing magazine **37**(6), 117–127 (2020).

[40] W. Hu, J. Pang, X. Liu, D. Tian, C.-W. Lin, and A. Vetro, "Graph signal processing for geometric data and beyond: Theory and applications," IEEE Transactions on Multimedia **24**, 3961–3977 (2021).

[41] L. Stanković, M. Daković, and E. Sejdić, "Introduction to graph signal processing," Vertex-Frequency Analysis of Graph Signals 3–108 (2019).

[42] P. Ferrer-Cid, J. M. Barcelo-Ordinas, and J. Garcia-Vidal, "Data reconstruction applications for iot air pollution sensor networks using graph signal processing," Journal of Network and Computer Applications **205**, 103434 (2022).

[43] K. Han, Y. Wang, J. Guo, Y. Tang, and E. Wu, "Vision gnn: An image is worth graph of nodes," Advances in neural information processing systems **35**, 8291–8303 (2022).

[44] Z. Guo and H. Wang, "A deep graph neural network-based mechanism for social recommendations," IEEE Transactions on Industrial Informatics **17**(4), 2776–2783 (2020).

[45] A. A. Duval, V. Schmidt, A. Hernández-García, S. Miret, F. D. Malliaros, Y. Bengio, and D. Rolnick, "FAENet: Frame averaging equivariant GNN for materials modeling," in *Proceedings of the 40th International Conference on Machine Learning*, PMLR.

[46] X.-M. Zhang, L. Liang, L. Liu, and M.-J. Tang, "Graph neural networks and their current applications in bioinformatics," Frontiers in genetics **12**, 690049 (2021).

[47] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NeurIPS* (2016).

[48] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations* (2017).

[49] A. Benamira, B. Devillers, E. Lesot, A. K. Ray, M. Saadi, and F. D. Malliaros, "Semi-supervised learning and graph neural networks for fake news detection," in *IEEE/ACM ASONAM* (2020).

[50] V. N. Ioannidis, A. G. Marques, and G. B. Giannakis, "A recurrent graph neural network for multi-relational data," in *IEEE ICASSP* (2019).

[51] F. Gama *et al.*, "Aggregation graph neural networks," in *IEEE ICASSP* (2019).

[52] S. Sharma and T. Varma, "Graph signal processing based underwater image enhancement techniques," Engineering Science and Technology, an International Journal **32**, 101059 (2022) https://www.sciencedirect.com/science/article/pii/S2215098621001804 doi: https://doi.org/10.1016/j.jestch.2021.09.005.

[53] J. A. Castro-Correa, J. H. Giraldo, A. Mondal, M. Badiey, T. Bouwmans, and F. D. Malliaros, "Time-varying signals recovery via graph neural networks," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2023), pp. 1–5, doi: 10.1109/ICASSP49357.2023.10096168.

[54] P. Xu, S. Xu, K. Shi, M. Ou, H. Zhu, G. Xu, D. Gao, G. Li, and Y. Zhao, "Prediction of water temperature based on graph neural network in a small-scale observation via coastal acoustic tomography," Remote Sensing **16**(4) (2024) https://www.mdpi.com/2072-4292/16/4/646 doi: 10.3390/rs16040646.

[55] J. A. Castro-Correa, J. H. Giraldo, M. Badiey, and F. D. Malliaros, "Gegenbauer graph neural networks for time-varying signal reconstruction," IEEE Transactions on Neural Networks and Learning Systems **35**(9), 11734–11745 (2024) doi: 10.1109/TNNLS.2024.3381069.

[56] E. Alcantara, L. Atlas, and S. Abadi, "Direction-of-arrival estimation using signal processing on graphs," in *2021 IEEE Statistical Signal Processing Workshop (SSP)* (2021), pp. 1–5, doi: 10.1109/SSP49050.2021.9513776.

[57] M. Li and X. Pan, "Direction of arrival estimation based on graph signal processing for a coprime array," in *OCEANS 2022, Hampton Roads* (2022), pp. 1–4, doi: 10.1109/OCEANS47191.2022.9977268.

[58] P. S. Wilson, D. P. Knobles, and T. B. Neilsen, "Guest editorial an overview of the seabed characterization experiment," IEEE Journal of Oceanic Engineering **45**(1), 1–13 (2020) doi: 10.1109/JOE.2019.2956606.

[59] P. S. Wilson, D. P. Knobles, and T. B. Neilsen, "Guest editorial: Continued exploration of fine-grained sediments from sbcex2017," IEEE Journal of Oceanic Engineering **47**(3), 497–502 (2022) doi: 10.1109/JOE.2022.3174252.

[60] "Bureau of Ocean Energy Management (BOEM) and National Oceanic and Atmospheric Administration (NOAA). MarineCadastre.gov. [AIS_2017_03_Zone19 and AIS_2017_04_Zone19]. Retrieved [August 07, 2020] from" , https://marinecadastre.gov/ais/.

[61] E. K. Westwood, C. T. Tindle, and N. R. Chapman, "A normal mode model for acousto-elastic ocean environments," The Journal of the Acoustical Society of America **100**(6), 3631–3645 (1996).

[62] J. Hildebrand, "Anthropogenic and natural sources of ambient noise in the ocean," Marine Ecology Progress Series **395**, 5–20 (2009).

[63] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," arXiv preprint arXiv:1904.08779 (2019).

[64] For GNNs, we extend the definition of graph signals to M-dimensional feature vectors so that we represent the input features as $\mathbf{X} \in \mathbb{R}^{N \times M}$.

[65] E. Isufi, F. Gama, D. I. Shuman, and S. Segarra, "Graph filters for signal processing and machine learning on graphs," arXiv preprint arXiv:2211.08854 (2022).

[66] H. Lu, L. Wang, X. Ma, J. Cheng, and M. Zhou, "A survey of graph neural networks and their industrial applications," Neurocomputing **614**, 128761 (2025) https://www.sciencedirect.com/science/article/pii/S0925231224015327 doi: https://doi.org/10.1016/j.neucom.2024.128761.

[67] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR-W* (2019).

[68] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, ICML*, Atlanta (2013), Vol. 3, p. 896.

[69] V. Kalofolias, "How to learn a graph from smooth signals," in *International Conference on Artificial Intelligence and Statistics* (2016).

[70] V. Kalofolias and N. Perraudin, "Large scale graph learning from smooth signals," in *International Conference on Learning Representations* (2018).