ICES Journal of Marine Science

International Council for the Exploration of the Sea
Conseil International pour l'Exploration de la Mer

ICES Journal of Marine Science (2021), https://doi.org/10.1093/icesjms/fsab140

Original Article

Semi-supervised target classification in multi-frequency echosounder data

Changkyu Choi [1], Michael Kampffmeyer [1,2], Nils Olav Handegard [3], Arnt-Børre Salberg [2], Olav Brautaset [2], Line Eikvil, [2], and Robert Jenssen [1,2,*]

[1] *UiT The Arctic University of Norway, P.O. Box 6050, Langnes Tromsø 9037, Norway*
[2] *Norwegian Computing Center, P.O. Box 114, Blindern, Oslo 0314, Norway*
[3] *Institute of Marine Research, Nordnesgaten 50, Bergen 5005, Norway*

Corresponding author: tel: (+47) 77646493; e-mail: robert.jenssen@uit.no

Choi, C., Kampffmeyer, M., Handegard, N. O., Salberg, A.-B., Brautaset, O., Eikvil, L., and Jenssen, R. Semi-supervised target classification in multi-frequency echosounder data . – ICES Journal of Marine Science, 00: 1–13.

Received 8 November 2020; revised 20 May 2021; accepted 8 June 2021.

Acoustic target classification in multi-frequency echosounder data is a major interest for the marine ecosystem and fishery management since it can potentially estimate the abundance or biomass of the species. A key problem of current methods is the heavy dependence on the manual categorization of data samples. As a solution, we propose a novel semi-supervised deep learning method leveraging a few annotated data samples together with vast amounts of unannotated data samples, all in a single model. Specifically, two inter-connected objectives, namely, a clustering objective and a classification objective, optimize one shared convolutional neural network in an alternating manner. The clustering objective exploits the underlying structure of all data, both annotated and unannotated; the classification objective enforces a certain consistency to given classes using the few annotated data samples. We evaluate our classification method using echosounder data from the sandeel case study in the North Sea. In the semi-supervised setting with only a tenth of the training data annotated, our method achieves 67.6% accuracy, outperforming a conventional semi-supervised method by 7.0 percentage points. When applying the proposed method in a fully supervised setup, we achieve 74.7% accuracy, surpassing the standard supervised deep learning method by 4.7 percentage points.

Keywords: acoustic target classification, deep clustering, limited annotation, pseudo-labeling, semi-supervised deep learning

Introduction

Acoustic target classification is a field of research that analyzes the marine acoustic data for the marine ecosystem and fishery management, and an analysis task of multi-frequency echosounder data is a major interest (Korneliussen, 2018). The goal is to assign an observed acoustic backscattering intensity to a given acoustic category. The results can be used to estimate the abundance or biomass of the species (MacLennan and Simmonds, 2013).

One common approach for acoustic target classification is manual categorization, where the operators identify and select regions with similar acoustic properties (Korneliussen, 2018). This manual categorization may be supported by relative frequency response (Kloser *et al.*, 2002; Korneliussen and Ona, 2003), echo traces (Reid, 2000), trawl sampling (Handegard and Tjøstheim, 2009), and domain knowledge of the target categories. However, the application of the supporting methods is limited due to their extremely high cost, making the manual process vulnerable to bias from the operators. Hence, automated and scalable analysis methods are required to efficiently cope with the multi-frequency data.

Deep learning, a family of data-driven computational models known for their flexibility and scalability, can provide an answer to the need. Especially convolutional neural networks (CNNs), a popular deep learning framework, are renowned to excel at image tasks (Long *et al.*, 2015). Although echosounder data are not images in the traditional sense, there exist commonalities between the two. Both data sources reflect visual observations, where each observation channel provides a structured form of the data in a two-dimensional array. Based on the commonality, a few studies have successfully applied the CNNs to perform target classification on the echosounder data, where the tasks are detection of sandeel (SE) schools (Brautaset *et al.*, 2020) and herring schools (Rezvanifar *et al.*, 2019). These CNNs learn how to extract abstract characteristics from patterns in the echosounder data, and the extracted characteristics are referred to as feature representation.

The feature representation that the neural networks learn is dependent on the formulated objective function. The objective function is designed to reflect the goal of the task, and measures an error between the current prediction of the CNN and the optimum that is often the human-provided annotation. "Fully supervised learning" refers to algorithms where the entire training data set is annotated. The learning scheme of the CNN is an iterative optimization process that gradually minimizes the error measured by the objective function. Provided a high quality of the training data and that an appropriate choice of the CNN are assured, the fully supervised learning approaches achieve a good level of performance as the model learns the feature representations in a way to mimic the corresponding annotations of the data.

It is, however, extremely costly and challenging to acquire the annotations in many real-world data including the echosounder data. The aforementioned acoustic target classification studies using CNNs learn in a fully supervised fashion, which heavily depends on the manual categorization process by the operators in order to train their models. Hence, new learning schemes are required in order to deal with an increasing volume of the datasets in an efficient and effective manner, where the dependency on the annotated data is reduced.

In this paper, we propose a novel deep learning algorithm for acoustic target classification, which operates on the condition that only a small part of the data is annotated, referred to as semi-supervised deep learning (Chapelle *et al.*, 2009). The novelty of our work is that the proposed algorithm exploits the underlying structure of the data including both the annotated part and the unannotated part using two interconnected objective functions, namely, a clustering objective and a classification objective. The alternating optimization process by the two objective functions allows the unannotated part of data to contribute to form decision boundaries with respect to the given classes, which is not applicable for a common supervised deep learning (SDL). To the best of our knowledge, this is the first semi-SDL algorithm applied for the acoustic target classification.

The multi-frequency echosounder data used in this study have been annually collected at the North Sea since 2009 by the Norwegian Institute of Marine Research for the case study of classifying lesser SE (*Ammodytes marinus*), a small fish without a swim bladder. Due to the abundance and fat richness (Raitt, 1934), it is considered as the major forage fish of the food chain, preyed on by a great variety of predators such as piscivorous fish species, marine mammals, and seabirds (Daan *et al.*, 1990; Furness, 2002). Analogously, the depletion of the SE stock causes a severe damage to the ecosystems (Johnsen *et al.*, 2017). For instance, Frederiksen *et al.* (2007) argue that there were exceptionally high breeding failures for most seabird species in the North Sea in 2004, due to a sharp decline of SE stocks in 2003, where the annual landing of SEs in 2003 was reduced to approximately 40% of the average landings in the ten previous years (ICES, 2017). The proposed method considerably reduces the dependency on the annotated data and contributes to the automated SE stock estimation, which is important for the ecosystems as well as the fisheries in the North Sea.

Extensive experiments conducted on this SE echosounder data validate the robustness of the proposed method. Regarding the patch-level semantic segmentation task, which classifies small and fixed-shaped patches extracted in a regular grid from the multi-frequency echosounder data, the proposed method outperforms both the semi-supervised benchmark under the partially annotated condition and the standard SDL under the fully annotated condition.

The contributions of this article are (i) to develop a novel semi-SDL algorithm that is suitable for segmenting and classifying echosounder data without prior information, and (ii) to demonstrate the proposed algorithm on a real test case.

Background and material

Echosounder data collection

In every April and May since 2005, The Norwegian Institute of Marine Research has conducted acoustic trawl surveys in the SE areas of the North Sea (Johnsen *et al.*, 2017). The SE echosounder data are measured during the surveys by multifrequency Simrad EK60 echosounder systems operating at four different frequency channels (18, 38, 120, and 200 kHz) on the vessel whose speed was approximately 10 knots. The echosounders were calibrated in accordance with the standard procedures before each survey. See Johnsen *et al.* (2009) for further details.

For each frequency channel, a volume backscattering coefficient s_v, an average amount of backscattering intensity per cubic metre (MacLennan *et al.*, 2002), is stored as a corresponding pixel value of the two-dimensional echosounder data. The data are collected at 1 Hz. The horizontal length of a single pixel is 1 second and the vertical length of a single pixel is 19.2 centimeters based on the pulse duration of 1.024 milliseconds. The height and width of the echosounder data, therefore, depends on the depth of the sea and the navigating time for the survey. We analyze echosounder data that have been collected between 2011 and 2019. The average height of the echosounder data is 399 pixels, corresponding to 76.6 meter depth. The total navigation time is 2,407 hours, which is approximately 11 days per year. For cross-validation, we split the data into two groups by year and assign the data between 2011 and 2017 to the training set, and the data from 2018 to 2019 to the test set.

Preprocessing and pixel-level annotation

In the preprocessing phase, all the volume backscattering values s_v are transformed in a decibel unit (dB re 1m^{-1}). The values less than -75 dB re 1m^{-1} or greater than 0 dB re 1m^{-1} are set to -75 dB re 1m^{-1} or 0 dB re 1m^{-1}, respectively. Infrequently, a few number of columns of the data are missing due to the temporary poor reception of the echosounder. We impute the minimum value -75 dB re 1m^{-1} to the missing columns with respect to a common time-range grid based on the resolution of the 200 kHz echosounder data. Pixels with NaN (not a number) are also replaced with -75 dB re 1m^{-1}. We leverage both pixel-level annotation and preprocessing methods from the earlier work (Brautaset *et al.*, 2020), for which we share the echosounder data.

Each pixel in the echosounder data is annotated into three classes based on the frequency response, where the classes are SE, other fish species (OT), and background (BG). An expert operator manu-

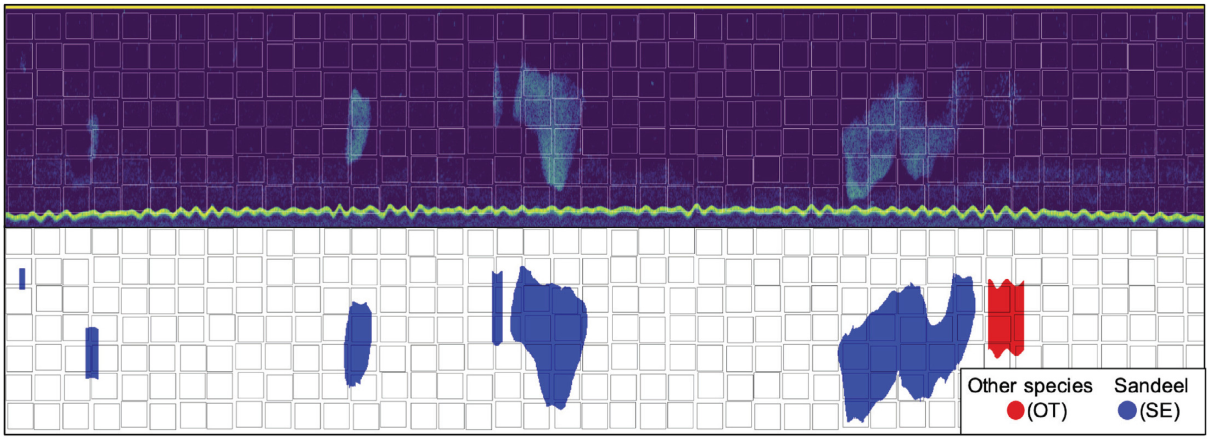


Figure 1. A part of the echosounder data at 200kHz (**up**) and the corresponding pixel-level annotation map (**down**). Each square indicates a patch of size 32 × 32 pixels, where the squares are regularly overlaid on the echosounder data with a random shift in a range of $[-2, 2]$ pixels both horizontal and vertical axes. Each echosounder patch and the corresponding annotation patch are extracted from the same location. Patches having a surface effect (yellow line at the top of the echosounder data) are discarded. SE are colored blue and the school of OT is red in the pixel-level annotation map.

ally delineates the fish school boundaries and annotates the schools across all years using the Large Scale Survey System software (Korneliussen *et al.*, 2016). The primary frequency for the software is chosen to 200 kHz considering the highest SE signal-to-noise ratio (Johnsen *et al.*, 2009). The operator adjusts the detection threshold centered at −63 dB at the primary frequency to visually distinguish the fish school boundaries. The delineated boundary is refined using binary morphological closing to have smoother and realistic edges (Brautaset *et al.*, 2020). The species decision process of the delineated fish schools is also manually performed by inspecting the frequency response for each detected school and is further validated by trawl samples where applicable. In addition to the expensive manual process, there is an element of tacit knowledge as with any expert system. This challenges to reliably define the criteria for the classification, as an effect from the operator may implicitly influence the decision.

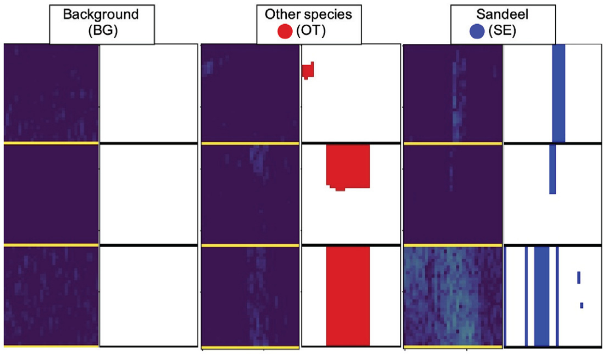


Figure 2. Nine pairs of the patches extracted from the echosounder data at 200 kHz and the corresponding pixel-level annotation map. Three patches are randomly selected per class of BG, OT, or SE.

Patch extraction and annotation

In general, CNN-based image tasks assume a fixed dimension of both an input image and the outcome. To apply CNN on the echosounder data, we extract fixed and small-sized patches from the data. Each extracted patch consists of 32 × 32 × 4 pixels, where “4” refers to the number of echosounder channels. This patch classification task can be seen as a down-stream task since the CNN learns visual features from the patches, and abstracts the learned features to class prediction vectors, where the length of the vector is equal to the number of classes to predict. Note that each element in the vector represents the probability of the class prediction of the patch with respect to each class that is achieved by the softmax function (see deep learning terminologies in the Appendix for the further details).

For the training patch extraction, we administer two criteria to avoid potential sources of bias: overlap between patches is not allowed, and the extracting location of a patch should be determined with stochasticity. Abiding by the criteria, we first overlay grid points spacing 36 × 36 on both the echosounder data and the corresponding pixel-level annotation map. Figure 1 depicts the overlay of the windows for patch extraction based on the grid points. Each grid point becomes the center of the window for the patch extraction, randomly shifted within a range of $[-2, 2]$ pixels to both width and height axes to add stochasticity. Due to the margin in the spacing of the overlaid grid points, there is no overlap between patches. Note that the stochastic spacing is only applied to the training set. The patches from the test set are extracted from a fixed grid, where the centroids are spaced in 32 × 32. To neglect the undesired surface effect that lies at the first ten rows from the top of each echosounder data, we locate the grid points in a way that patches exclude this surface effect. Figure 2 shows the patches from the echosounder data and the pixel-level annotation map.

We annotate each echosounder patch leveraging the corresponding pixel-level annotation. According to the extracted patch dimension, 1024 (32 × 32) annotated pixels determine the patch annotation. We assign the SE or OT class to the patch, where the number of corresponding fish pixels is greater than or equal to 16 pixels which occupy 1.56% of the pixels in the patch. On the other hand, the patch without fish-annotated pixels is annotated to the BG class. The number of patches having both SE and OT pixels together or one fish class but less than 16 fish pixels is negligibly small and those patches are discarded.

Table 1. Extracted patches from the training echosounder data (2011–2017), and the test echosounder data (2018–2019).

Year	Training set (2011–2017)		Test set (2018–2019)	
Class	Extracted patches (percentage)	Undersampled patches	Extracted patches (percentage)	Undersampled patches
BG	1 200 075 (97.81)	10 922	816 726 (97.61)	6 004
OT	15 965 (1.30)		6 004 (0.72)	
SE	10 922 (0.89)		13 984 (1.67)	
Total	1 226 962 (100.00)	32 766	836 714 (100.00)	18 012

Table 1 represents the number of patches extracted from the echosounder data. Severe class imbalance is observed, with more than 97% of the patches belonging to the BG class. To tackle the class imbalance, we randomly undersample patches from the majority classes to obtain the same number of patches for each of the classes (Buda *et al.*, 2018), resulting in a total number of training patches of 32766, and a total number of test patches of 18012. The patches that are excluded from both the training set and the test set are leveraged for tuning hyperparameters.

Deep clustering

We present a novel semi-SDL method, where the idea of the proposed method is to exploit *both* the intrinsic structure of the data and the available annotation, in a single CNN. This method can be applied to the echosounder data as well as being potentially generalized to other data sources since it incorporates the generic idea of deep clustering into the SDL.

Deep clustering refers to unsupervised deep learning based approaches, that aim to cluster data into underlying groups without requiring the class attributes of the data (Korneliussen, 2018). It leverages the representation power of the neural network in conjunction with clustering algorithms, and partitions the input data into clusters with respect to the learned representation. As clustering performance heavily depends on the underlying structure of the data, deep clustering leverages the neural network to encode the training images in the feature representations where the clustering task becomes much easier (Jabi *et al.*, 2019).

There are two main directions of deep clustering with respect to designing the objective function, namely, cluster-discriminative and cluster-generative objectives. Using mutual information or divergence measures, models with cluster-discriminative objectives learn the decision boundaries in-between clusters via posteriors over the assignments given the inputs (Jabi *et al.*, 2019). Deep divergence-based clustering (DDC) exemplifies this line of research (Kampffmeyer *et al.*, 2019), where the objective of DDC is designed to increase divergence between clusters while achieving compactness within a cluster using information-theoretic divergence measures. Deep clustering models that utilize cluster-generative objectives, such as k-means, have also been studied (Caron *et al.*, 2018; Biernacki *et al.*, 2000). In their model, referred to as DeepCluster, they explicitly model the density of datapoints within the clusters via likelihood functions. For a given image dataset, the k-means clustering models K different densities, where each density refers to an image descriptor or a visual feature. This has the advantage that it is easy to increase the capacity of more visual features by simply increasing the number of clusters K, leading to all-purpose visual features.

The scalability of the visual features in the DeepCluster is the reason why our method takes its main inspiration from Caron *et al.* (2018) when analyzing the echosounder data. The echosounder patches have many sources that can cause a large variance within their feature representations. Examples include the type of fish, the arrangement and density of the fish pattern, and the location and the occupied area of the fish pattern inside the patch, to name a few. The method of Caron *et al.* (2018) enables to partition the feature representations across the numerous sources of the variance into many clusters, and eventually discovers the intrinsic structure of the data.

However, there is potentially valuable information given by even just having a few annotations and it is crucial to be able to leverage this information. Hence, we propose a new approach that has the capability to also exploit annotated data, even in small amounts.

Method

Objective functions

The key novelty of this paper is to propose a new type of deep neural network leveraging vast amounts of unannotated data (unsupervised) while being able to simultaneously exploit some available annotated data (supervised), yielding a novel *semi-SDL* algorithm. This is achieved through the optimization of an unsupervised clustering objective in addition to a supervised classification objective as outlined in Figure 3. The alternating optimization process enables a CNN that is trained through two interconnected objective functions.

The clustering objective, which utilizes ideas from the study of Caron *et al.* (2018), exploits the underlying structure of the data using k-means without requiring any annotation. *The classification objective* enforces consistency of predictions with regards to the given classes in the annotated data. These objectives optimize the CNN in an alternating manner. Through our alternating optimization procedure, we further indirectly incorporate the annotation information into the model, influencing the clustering objective to learn both a structured representation as well as a representation that is consistent with the available annotations. Figure 4 outlines the learning procedure that is further described below.

Clustering objective

Refer to the Appendix for detailed information of the terminologies, such as a cross-entropy loss, end-to-end learning, softmax, and epoch. The clustering objective of our proposed semi-supervised model aims to address both the clustering of the input data as well as the optimization of the CNN.

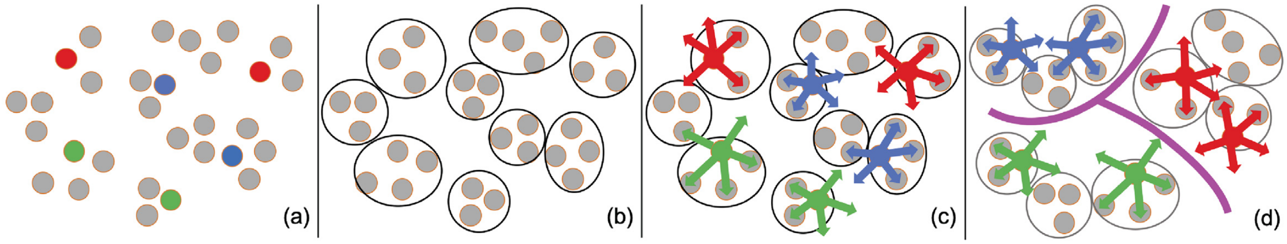


Figure 3. Overview of the proposed method. Each point represents the extracted patch, where the point in gray is unannotated while the points in color (red, green, or blue) indicates the annotated one with respect to the class. (a) The training data occupy an arbitrary space. (b) The clustering objective helps to form clusters regardless of the annotation. (c) The available annotated data and the classification objective optimize the CNN in a supervised manner. (d) The iteration of (b) and (c) constructs the decision boundary with respect to given classes, where the unannotated points take their place inside the boundary according to their own clusters.

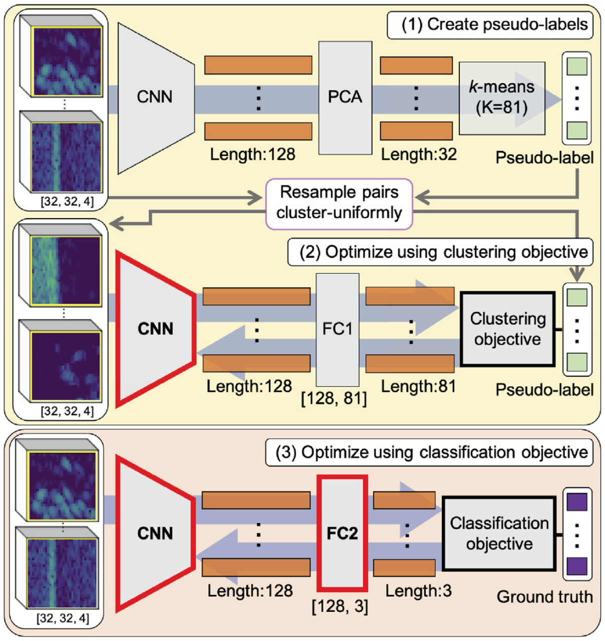


Figure 4. Training procedure of the proposed method. Each orange bar represents the feature representation of each patch in a vectorial form of the specified length. We configure that the output of the CNN is a vector of length 128. Only the CNN and FC2 layer with red outlines are optimized in each stage. (1) Create pseudo-labels. (2) Optimize the CNN using cluster objective. (3) Optimize the CNN and FC2 using classification objective.

The proposed method takes inspiration from the study of Caron *et al.* (2018) that clusters using k-means and optimizes the CNN based on the cluster assignments, which are called pseudo-labels. The proposed method clusters the feature representations of all training patches into K clusters using k-means, in a way to find the best assignment that minimizes the k-means loss:

$$\mathcal{L}_{kmns} = \frac{1}{N}\sum_{i=1}^{N} \min_{\mathbf{c}_k} d(\mathbf{h}^{(i)}, \mathbf{c}_k). \tag{1}$$

In this expression, N is the number of training patches, $d(\cdot, \cdot)$ is the L_2 distance between two vectors, $\mathbf{c}_k$ is the centroid of the cluster k, $\mathbf{h}^{(i)} = g\big(f_\theta(\mathbf{x}^{(i)})\big)$ are the principal components of the feature representations of the i^{th} input training patch $\mathbf{x}^{(i)}$, $f_\theta(\,\cdot\,)$ is the CNN that produces the feature representation, and $g(\,\cdot\,)$ computes principal component analysis (PCA). Note that we perform PCA (Wold *et al.*, 1987) on the feature representations before clustering in order to use only the first few principal components for manageable computational complexity. Also note that the CNN remains fixed without being optimized in this step.

Next, we optimize the CNN to learn the feature representations clustered by k-means. The CNN is trained in a supervised manner by the supervision of the pseudo-labels, not the annotations, where the assignment indices from the result of the k-means clustering become the pseudo-labels. A cross-entropy loss, which is a standard choice for the classification task in SDL, is used for the optimization.

To align the lengths of the feature representation and the pseudo-label to K, we append a single fully connected (FC) layer with a softmax at the end of the CNN, depicted as FC1 in Figure 4. The CNN appended by FC1 becomes an end-to-end learning model.

The clustering objective is depicted as:

$$\mathcal{L}_{cls} = \frac{1}{N}\sum_{i=1}^{N} CE\{\tilde{f}_\theta(\mathbf{x}^{(i)}), \hat{\mathbf{y}}^{(i)}\}, \tag{2}$$

where $CE(\mathbf{z}, \mathbf{y}) = -\sum_k y_k \log(z_k)$ is the cross-entropy loss of a single datapoint, $\hat{\mathbf{y}}^{(i)} \in \{0, 1\}^K$ is the one-hot encoded pseudo-label of $\mathbf{x}^{(i)}$, and $\tilde{f}_\theta(\cdot)$ is the FC1-appended CNN that produces the pseudo-label prediction. The entire set of the pseudo-labels is changed each time when a new clustering result is obtained. We randomly initialize the weights of FC1, which aligns the representation of the CNN to the pseudo-labels, for each new update of the pseudo-label set.

Classification objective

The classification objective enforces consistency of predictions with regard to the given classes in the partially available annotated data. Using available annotated data, we train the model in a supervised manner with respect to the given classes, anticipating that the model learns the feature representations to compact each cluster in terms of the annotated data. The learned representations are reflected in updating the clustering structure, in such a way that the structure converges with respect to the given class distribution. Note that the class indices matter in this step. After removing FC1 from the CNN, we append another FC layer with softmax, called FC2, at the same place, to learn the class prediction using the cross-entropy loss. The CNN appended by FC2 also becomes an end-to-end learning model.

The classification objective is depicted as:

$$\mathcal{L}_{sup} = \frac{1}{L}\sum_{i=1}^{L} CE\{\ddot{f}_\theta(\mathbf{x}^{(i)}), \mathbf{y}^{(i)}\}, \tag{3}$$

where $L \leq N$ represents the number of annotated data, C represents the number of classes to predict, $\mathbf{y}^{(i)} \in \{0, 1\}^C$ represents the annotation of $\mathbf{x}^{(i)}$, and $\ddot{f}_\theta(\cdot)$ represents the FC2-appended CNN that produces the class prediction.

Training procedure

The combined optimization leveraging both the clustering objective and the classification objective in the single CNN constitutes a novel semi-SDL method. The training procedure consists of three stages: (1) create pseudo-labels using k-means; (2) optimize the model using the clustering objective; and (3) optimize the model using the classification objective. The iteration of the stages from (1) to (3) optimizes the CNN. Figure 4, Algorithm 1, and Algorithm 2 illustrate the procedures.

(1) Create pseudo-labels using k*-means*

The CNN provides the feature representations by processing all training patches. These principal components of the feature representations processed by PCA are clustered to K clusters by k-means as shown in Equation (1). The cluster index of each patch becomes a pseudo-label. This stage is done when each patch in the training set has its cluster index that implies the clustering structure. The CNN processes the patches but is not optimized in this stage.

(2) Optimize the model using the clustering objective

This stage aims to optimize the CNN under the supervision of the pseudo-labels. We first construct the pairs consisting of the patch and the pseudo-label. The pseudo-labels should be cluster-balanced to avoid the trivial solutions of the k-means (Yang *et al.*, 2017). To enforce this balance, we sample pairs from each cluster up to the average number of patches per cluster. Replacement is tolerated if the cluster does not have enough pairs in it with respect to this average number of patches per cluster. We append FC1 and train the CNN in an end-to-end manner with these uniformly sampled pairs, where FC1 has weights which maps the feature representations before PCA to K clusters, and zero bias as depicted in Equation (2). The CNN is optimized by the gradients that backpropagates via FC1. Note that FC1 is not optimized as the cluster indices are randomly changeable. Instead we initialize the parameters in FC1.

(3) Optimize the model using the classification objective

This stage aims to learn by the supervision of a few available class-wise annotations (three classes in our case study). FC1 is removed from the end of the CNN, and FC2 with zero bias and the weight that maps the feature representations to given labeled classes is appended. Note that we keep the parameters of FC2 from the previous turn to maintain consistency of the class prediction.

This provides another end-to-end learning model that is supervised by the annotation of three classes as shown in Equation (3). The model including the CNN and FC2 is updated with gradient decent. The prediction accuracies of the training set is measured after the optimization. For the next iteration, we remove FC2 after exporting the weight values and repeat the stage **(1)**.

Algorithm 1 Create pseudo-labels using k-means

```
Input: training patches $\mathcal{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$
Output: pseudo-labels $\hat{\mathcal{Y}} = \{\hat{\mathbf{y}}^{(i)}\}_{i=1}^{N}$
Procedure:
  while $i \neq N$ do
    Process $\mathbf{x}^{(i)}$ to the CNN $f_\theta$
    Reduce dimension of $f_\theta(\mathbf{x}^{(i)})$ using PCA and store
  end while
  Cluster the stored feature representations using $k$-means
  Create pseudo-label $\hat{\mathbf{y}}^{(i)}$ using the cluster assignment of $\mathbf{x}^{(i)}$
```

Algorithm 2 Optimize the model by alternating two objectives

```
Input: $\mathcal{X}$, $\hat{\mathcal{Y}}$ from Algorithm 1 and class annotation $\{\mathbf{y}^{(i)}\}_{i=1}^{L \leq N}$
Procedure:
  Sample the same number of $(\mathbf{x}^{(i)}, \hat{\mathbf{y}}^{(i)})$ pairs w.r.t pseudo-label
  Append randomly initialized FC1 at the end of CNN: $\tilde{f}_\theta$
  while $i \neq N$ do
    Process $\mathbf{x}^{(i)}$ to the CNN $\tilde{f}_\theta$
    Compute loss with (??) and update $\Theta$ with gradient descent except FC1
    if $\mathbf{y}^{(i)}$ exists then
      Replace FC1 to FC2: $\ddot{f}_\theta$
      Process $\mathbf{x}^{(i)}$ to the CNN $\ddot{f}_\theta$
      Compute loss with (??) and update $\Theta$ with gradient descent
    end if
  end while
```

Organizing training data for semi-SDL

The semi-supervised method we propose exploits both the data structure in the entire set of training patches as well as in a few annotated patches.

Under the assumption that the total number of the patches is fixed, data organization for the method is characterized by the annotation ratio, which indicates the ratio of the annotated patches to the entire set of training patches. We set the total number of the training patches to 32766, and the total number of the test patches to 18012 as depicted in Table 1.

Annotation ratio

To construct the training data for the proposed method, we introduce the annotation ratio, which measures the ratio of the number of annotated patches to the number of the entire set of training patches. Four ratios are studied, namely, 1.000, 0.100, 0.050, and 0.025, where the annotation ratio of 1.000 represents full supervision. Table 2 illustrates the number of annotated and unannotated patches for each annotation ratio, where the number of unannotated patches is the same over the classes as we annotate patches according to the annotation ratio from the undersampled training patches. We refer this as unannotated-balanced (U-Ba), since the unannotated part is class-balanced. Figure 5 depicts the t-SNE plots of U-Ba with the annotation ratio of 0.100 case.

Table 2. The number of training patches for U-Ba case with respect to the classes BG, SE, and OT, and the annotation ratio.

Anno. ratio	**1.000**		**0.100**		**0.050**		**0.025**	
U-Ba	**Anno.**	**Unanno.**	**Anno.**	**Unanno.**	**Anno.**	**Unanno.**	**Anno.**	**Unanno.**
BG	10 922	0	1092	9830	546	10 376	273	10 649
OT	10 922	0	1092	9830	546	10 376	273	10 649
SE	10 922	0	1092	9830	546	10 376	273	10 649
Total	32 766	0	3276	29 490	1638	31 128	819	31 947
	32 766		32 766		32 766		32 766	

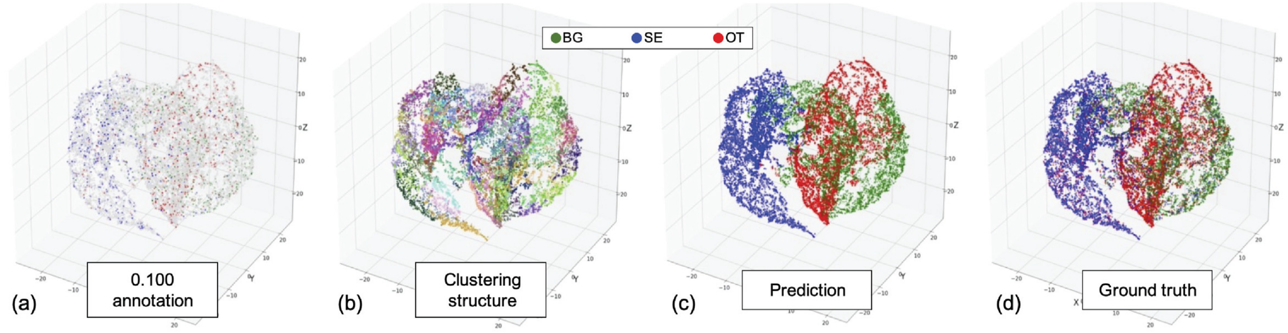


Figure 5. Three-dimensional t-SNE plots of the training patches (U-Ba, 0.100). (a) The distribution of training patches in an arbitrary space. Colored points represents the annotated patches, while gray points are unannotated ones. (b) Clustering structure of 81 clusters. The color differentiates the cluster assignment. (c) Class prediction. (d) The ground truth of the prediction.

Preserving class imbalance in unannotated part

It is important for the deep learning model to have a class-balanced training dataset since the imbalance of the data may cause bias that harms the generalization of the model prediction (Goodfellow *et al.*, 2016). To comply with this rule of thumb, we set the annotated part of the data to be class-balanced. However, when it comes to the unannotated part of the data, the rule of thumb is not applicable since the annotations are not accessible to know whether it is balanced or not.

The impact of the class imbalance in the unannotated part should be independently considered as this may potentially affect the performance of the proposed method. From our extracted patches, we observe the severe class imbalance. As shown in Table 1, 97.81% of the patches belong to the BG class.

To measure the robustness of the proposed method against the class imbalance in the unannotated part of the data, we institute a new setting referring to as unannotated-imbalanced (U-Im) in addition to U-Ba, where U-Im simulates the intrinsic class distribution before undersampling patches. Table 3 specifies the number of patches for the U-Im case. Note that the annotated part and the total number of patches are the same for those two cases.

Experiments

The purpose of the experiment on our SE case study is to explore the robustness of the proposed method in the semi-supervised learning environment that exploits limited annotations and, at the same time, the contribution of the unannotated data. In the experiments, we observe the prediction accuracy of the proposed method with different settings of the training set in terms of the annotation ratio and the unannotated data.

Unannotated data

Two settings for the unannotated part, U-Ba and U-Im, are suggested above. In parallel, to measure the lowerbound performance of the proposed model in terms of the unannotated data, we construct additional training sets that use only the annotated part of the data which is class-balanced, referred to as annotated only (AO). The number of patches over the classes is given in Tables 2 and 3. For example, with the annotation ratio of 0.025, the training set for AO case consists of 819 annotated patches without any unannotated patches. An annotation ratio of 1.000 is included in order to estimate the upperbound of the proposed method, where the model exploits full supervision of the annotations, while simultaneously learning the structure with the clustering objective.

Model description

We create our own CNN based on the architecture of VGG-16, but modify a few points including the input layer to utilize the four-channel patches in our CNN architecture.

The VGG-16 can be broadly divided into two parts, a feature extractor and a classifier. The feature extractor consists of in total 18 layers, 5 max-pooling layers with 2 × 2 kernels and 13 convolution layers with 3 × 3 filters, where the max-pooling layers are located in the 3^{rd}, 6^{th}, 10^{th}, 14^{th}, and 18^{th} layers. The remaining layers are convolution layers. Each convolutional layer is followed by batch normalization (Ioffe and Szegedy, 2015) and a rectified linear unit (ReLU) activation (Nair and Hinton, 2010). Based on the location of the pooling layer, the number of filters for each convolution layer

Table 3. The number of training patches for U-Im case with respect to the class and the annotation ratio.

Anno. ratio	**1.000**		**0.100**		**0.050**		**0.025**	
U-Im	**Anno.**	**Unanno.**	**Anno.**	**Unanno.**	**Anno.**	**Unanno.**	**Anno.**	**Unanno.**
BG	10 922	0	1092	28 845	546	30 446	273	31 248
OT	10 922	0	1092	383	546	405	273	415
SE	10 922	0	1092	262	546	277	273	284
Total	32 766	0	3276	29 490	1638	31 128	819	31 947
	32 766		32 766		32 766		32 766	

The unannotated part is shared according to their intrinsic distribution such that BG, OT, and SE classes occupy 97.81%, 1.30%, and 0.89%, respectively.

varies in 5 steps, where the first 2 layers have 64, the 4^{th} and the 5^{th} layers have 128, the layers from the 7^{th} to the 9^{th} have 256, and the layers from the 11^{th} to the 13^{th} and the 15^{th} to the 17^{th} have 512 filters.

We leverage the feature extractor part of VGG-16 with a modification of the input layer. Due to 5 max-pooling layers with with 2×2 kernels, the model reduces the dimension of the input patches to $1/2^5$, and the feature representations before the classifier have the vectorial form of $(1 \times 1 \times 512)$ that can be input to the classifier without flattening.

The classifier of VGG-16 has three FC layers with ReLU activation. To remove the effect from ReLU before k-means clustering, the last ReLU activation is discarded when the output of the classifier is supposed to be used for PCA. For regularization, dropout ($p = 0.5$) (Srivastava *et al.*, 2014) is performed after the first and second activation function in the classifier. The number of neurons for each layer is 4096, 4096, and 128, respectively. The outcome for the echosounder patch is set to a vector of length 128 considering the balance between the computational complexity and the available computing resources.

Training configuration

The model is trained by the use of mini-batch training, where the batch size is set to 32. The Adam optimizer (Kingma and Ba, 2015) with learning rate 3×10^{-5}, beta (0.9, 0.999), and weight decay 10^{-5} is applied for the all experiments. The three-stage training shown in Figure 4 is iterated 1000 times for all experiments, applying early stopping (Prechelt, 1998) on the condition that the accuracy is not improved for 100 times. We choose the first 32 principal components in Equation (1) as they capture most of the variance of the data. The training procedure for the proposed method is shared for all experiments. As discussed in the study of Caron *et al.* (2018), the choice of the number of the clusters K does not have a significant impact on the performance if we cluster the feature representations with a sufficiently large number of clusters compared to the number of classes. We have tested a set of different Ks, and choose K to be 81 considering the following reasons. (i) Classifying the patches up to $C = 3$ classes, we expect K to be expressed in terms of the number of classes C, such as $K = C^4$, expecting that each class has approximately C^3 clusters for the U-Ba case. (ii) Considering the total number of training patches $N = 32766$, the average number of patches in a cluster is approximately 400. Under the scenario of an annotation ratio of 0.025, each cluster has approximately 10 annotated patches. We tune those hyperparameters using the patches that are excluded from the training set and the test set in the undersampling process. All the codes are implemented in PyTorch (Paszke *et al.*, 2017).

Validation methods

For the validation of the proposed method, we introduce two baseline models to compare the performance. The first baseline is introduced to compare the performance of our deep learning method to a robust semi-supervised machine learning algorithm. We utilize the advanced semi-supervised support vector machine (S3VM) (Bagattini *et al.*, 2017), a statistical learning framework that is frequently used in many real-world applications. The S3VM classifier is trained based on the learned feature representations of length 128 from the proposed model using the radial basis function kernel for this non-linear classification problem.

The second baseline allows us to investigate the impact of the clustering objective in a supervised condition. The AO settings play this role. The proposed method that utilizes two objectives is compared with a common SDL model that leverages the classification objective only. The number of training patches for the AO settings depend on the annotation ratio as shown in Tables 2 and 3, where the patches are class-balanced. For the common SDL model, the entire training settings including the CNN architecture and related hyperparameters are shared with the proposed method in a supervised manner.

Results

Here, we focus mainly on the results form the class-balanced test set, as it demonstrates an impartial performance comparison that is not affected by the large class-imbalance.

For the class-balanced test case, the prediction accuracies for our SE case study within acoustic target classification as well as the F1 scores are presented in Table 4, where the best results are highlighted in bold. Overall, for the semi-supervised settings such as U-Im and U-Ba, the proposed model outperforms the semi-supervised benchmark S3VM (Bagattini *et al.*, 2017), and for the supervised settings referred to as AO, the proposed model achieves improved or comparable prediction performance compared to the standard SDL models over the entire set of annotation ratios. Figure 6 visualizes the prediction of the proposed method using t-SNE plots (Van der Maaten and Hinton, 2008).

Supervised case

Comparing the proposed method (ours) with the standard SDL under the AO setting with an annotation ratio of 1.000, ours (accuracy

Table 4. Prediction accuracies and F1 scores for the class-balanced test set.

Class-bal.	Accuracy						F1 score (three classes, macro averaging)					
test set	Semi-supervised				Supervised		Semi-supervised				Supervised	
Annotation	U-Im		U-Ba		AO		U-Im		U-Ba		AO	
ratio	Ours	S3VM	Ours	S3VM	Ours	SDL	Ours	S3VM	Ours	S3VM	Ours	SDL
1.000					**0.8202**	0.8000					**0.8190**	0.7966
0.100	**0.7814**	0.7341	**0.7896**	0.7340	**0.7531**	0.7496	**0.7794**	0.7313	**0.7872**	0.7313	**0.7481**	0.7462
0.050	**0.7484**	0.6668	**0.7694**	0.6668	0.6899	**0.6909**	**0.7447**	0.6653	**0.7666**	0.6654	0.6840	**0.6886**
0.025	**0.7364**	0.5838	**0.7159**	0.5827	**0.6495**	0.6108	**0.7326**	0.5774	**0.7153**	0.5765	**0.6468**	0.6109

S3VM (Bagattini *et al.*, 2017) and the standard SDL models are introduced as the benchmarks. The prediction accuracies and F1 scores of the test set are presented with respect to the settings of the training set.
Bold values denote statistical significance at the $p < 0.05$ level.

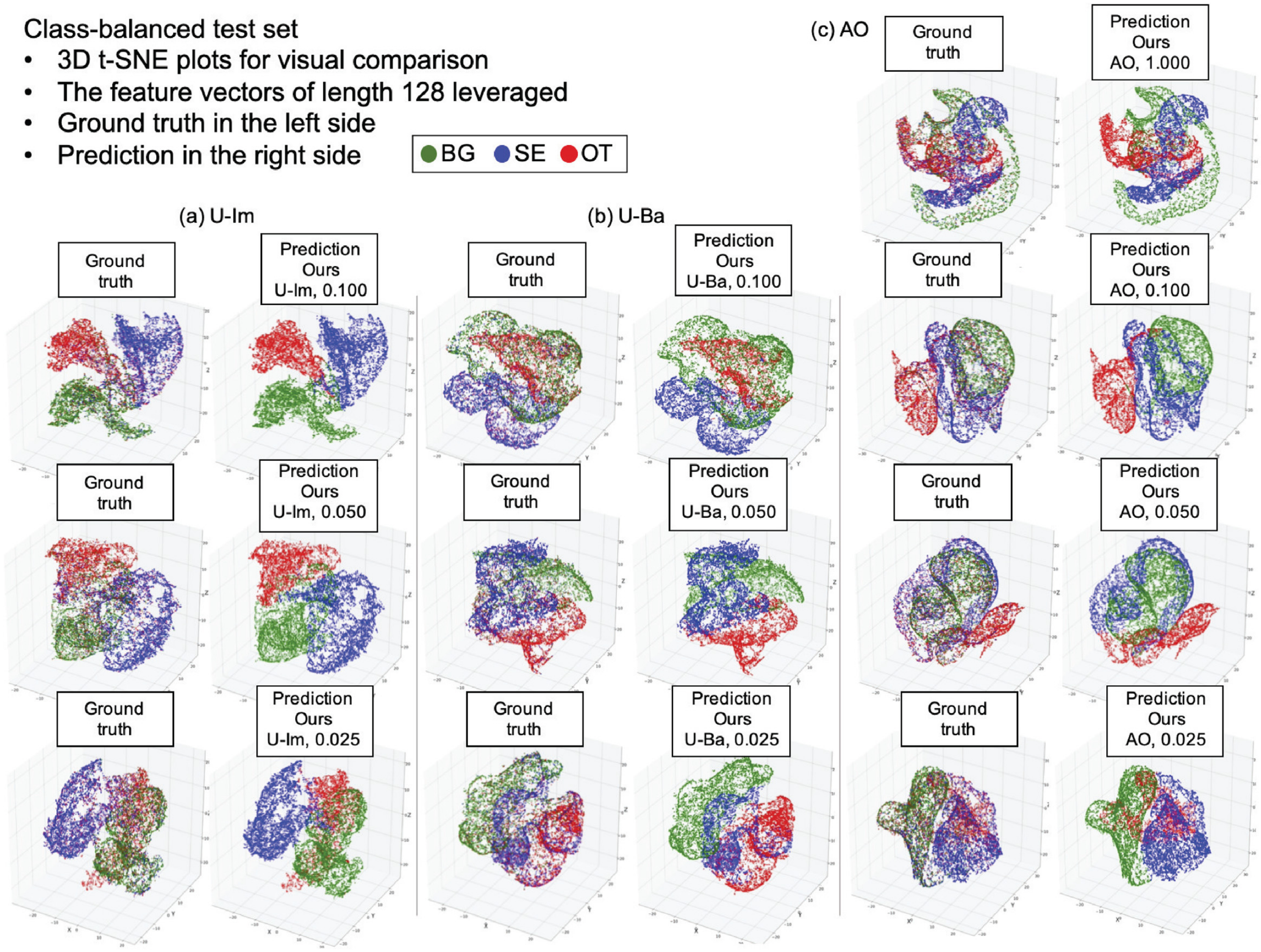


Figure 6. t-SNE plots for visual comparison (class-balanced test set). The feature vectors of the CNN for each setting are compressed for the three-dimensional plot. (a) U-Im, (b) U-Ba, and (c) AO. Less difference between the ground truth and prediction is observed from the higher annotation ratio.

0.8202) outperforms the standard SDL (accuracy 0.8000) by 2.02 percentage points. This trend is consistent also with other annotation ratios.

These results validate that the proposed method leveraging the unsupervised clustering objective improves the prediction performance over common SDL. We argue that the alternating optimization of the two proposed objectives leads the model to understand more about the global data distribution, and this contributes to creating improved decision boundaries compared to the traditional supervised learning approach that learns to mimic the given class attributes in the training set.

Annotation ratio

Throughout the cases, we observe as a tendency that the prediction accuracy increases as the annotation ratio increases. Interestingly, there is only 1.86 percentage points difference in accuracy between the proposed model with U-Im with 0.100 annotation ratio (U-Im, accuracy 0.7814) and the standard SDL with 1.000 annotation ratio (SDL, accuracy 0.8000), where the proposed method leverages a tenth of the annotated data against the standard SDL setting. The proposed method also outperforms the same annotation ratio (0.100) case of the standard SDL (accuracy 0.7496) by 3.18 percentage points.

The results indicate that the proposed method can effectively exploit a small amount of annotated data, and, to a certain extent, approximate the decision boundaries that are achieved by the fully SDL. We argue that, in the proposed method, the annotated data are leveraged by two different objectives respectively, which facilitate the interconnection of the two objectives in order to make good use of the annotated data. In this process, the unannotated data in a cluster gradually share the annotations that originate from the annotated data in the same cluster or the clusters nearby located, and eventually, the entire data in the same cluster have the same class prediction.

Class imbalance in unannotated data

In our method, the utilization of the unannotated data, found in the U-Im and U-Ba cases, considerably improves the prediction performance compared to the AO case under the same annotation ratio. In particular, the U-Im is comparable to the U-Ba setting. This includes the case where we in the U-Im setting (accuracy 0.7364) achieve 2.05 percentage points higher accuracy compared to the U-Ba setting (0.7159) with 0.025 annotation ratio. Those are promising results as a severe class imbalance is observed in the unannotated data for the U-Im case.

Confusion matrices

Figure 7 depicts the confusion matrices of the class-balanced test set, with respect to the annotation ratio and the unannotated part of the training set. For each matrix, the class BG is represented by the first row/column, the class SE is represented by the second row/column, and the class OT can be found in the third row/column. Each true class consists of one row and the probabilities of each row sums to one.

When comparing the diagonal components of the two confusion matrices for the semi-supervised cases, the proposed method can be seen to outperform the benchmark for all the classes and settings except two cases for the OT class in the U-Ba setting, where the accuracies are comparable (ours: 0.7840, S3VM: 0.7916 with the annotation ratio of 0.100, and ours: 0.7350, S3VM: 0.7465 with the annotation ratio of 0.025). Also, the degree of improvement is greater in the SE and BG classes than in the OT class. We believe that the reason for this is that the training patches in the SE and BG classes are more uniform than the ones in the OT class, which capture the backscattered response from diverse fish species when collected, and that deep clustering takes advantage of the uniformity when investigating the structure of the data.

We observe that the BG class achieves higher accuracy than the other classes, probably since the backscattering intensities in the BG patches are considerably more uniform, mostly having the lowest intensity. The SE class shows the lowest accuracy among the classes (e.g. 0.6755, U-Im with annotation ratio of 0.100), resulting in a higher false-negative rate (0.3245) and lower false-positive rate (0.1604).

This means that the predicted amount of SE will be a conservative estimate as the SE patches are frequently misclassified to other classes but the patches in the other classes are rarely misclassified to the SE class. We do not observe a tendency for any bias towards one class over the other for the misclassified SE patches.

The proposed method achieves more consistent performance against the variation of the annotation ratios compared to the benchmark in the semi-supervised cases. We argue that the proposed method is robust even the available annotated data are extremely few, as it approximates the relatively accurate decision boundary for the prediction by understanding the global distribution of the data, along with learning how to effectively exploit the available annotated data.

Class-imbalanced test set

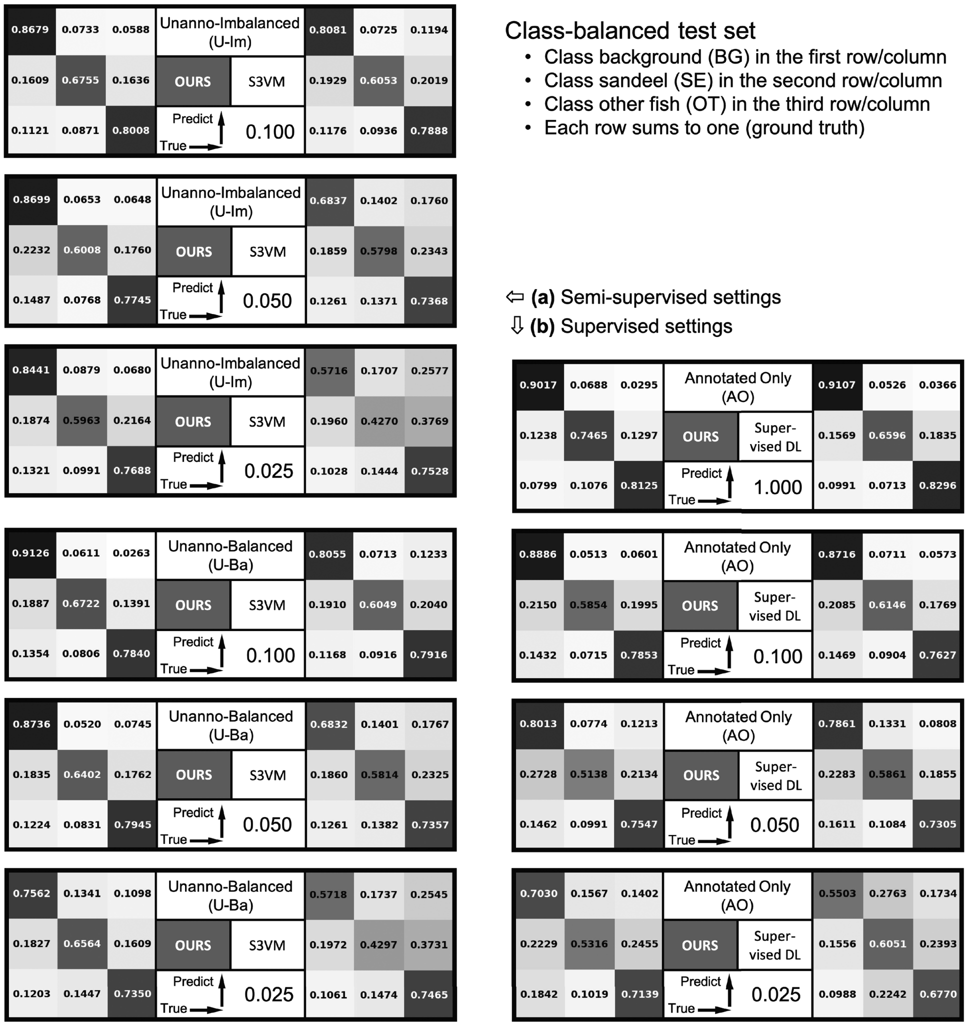
For the class-imbalanced test case, the prediction accuracies and the F1 scores are presented in Table 5, where the best result is highlighted in bold. Note that severe class imbalance causes bias in the result to a certain degree, where 97.61% of the test patches belong to the BG class as depicted in Table A1 in the Appendix. Overall, we observe the similar tendency that we discover from Table 4, where the proposed method outperforms the semi-supervised benchmark. Confusion matrices for the class-imbalanced test case is shown in Figure A1 in the Appendix.

For the class-imbalanced test case, the prediction accuracies and the F1 scores are presented in Table 5, where the best result is highlighted in bold. Note that the severe class imbalance causes a bias in the result as 97.61% of the test patches belong to the BG class as depicted in Table A1. Overall, we observe a similar tendency to what we discover from Table 4, where the proposed method outperforms the semi-supervised benchmark. Confusion matrices for the class-imbalanced test case are shown in Figure A1 in the Appendix.

Conclusion

In this paper, we proposed a novel semi-SDL method for acoustic target classification, which (ii) takes advantage of the power of deep learning, (ii) is trainable end-to-end in both semi-supervised and fully supervised manners, (iii) exploits the underlying structure of the training data regardless of the annotation, (iv) is robust against the class imbalance of the unannotated part of the data, and (v) achieves results that outperform or are comparable with other methods including a common SDL model. We have also investigated the performance through extensive experiments to evaluate the robustness of the method using rigorous criteria and compare the results with the advanced machine learning benchmark model. In addition, we have established a data organization process for semi-supervised learning to tackle the challenge of class imbalance. Overall, the promising results imply that the proposed method including the data organization process can be broadly applied to the severely class-imbalanced data with limited annotations, which are often found in the real world. To the best of our knowledge, this is the first semi-SDL paper in acoustic target classification.

In future work, we intend to explore other types of deep neural networks architectures beside the VGG-16 network. It would also be of interest to study other types of acoustic target classification problems. As a further example of future work, we intend to extend our method in order to categorize a single intensity of the multi-frequency echosounder data. This is known as pixel-level semantic segmentation, which potentially can contribute to



Class-balanced test set
- Class background (BG) in the first row/column
- Class sandeel (SE) in the second row/column
- Class other fish (OT) in the third row/column
- Each row sums to one (ground truth)

⇦ **(a)** Semi-supervised settings
⇩ **(b)** Supervised settings

0.8679	0.0733	0.0588	Unanno-Imbalanced (U-Im)		0.8081	0.0725	0.1194
0.1609	0.6755	0.1636	OURS	S3VM	0.1929	0.6053	0.2019
0.1121	0.0871	0.8008	Predict ↑ True →	0.100	0.1176	0.0936	0.7888

0.8699	0.0653	0.0648	Unanno-Imbalanced (U-Im)		0.6837	0.1402	0.1760
0.2232	0.6008	0.1760	OURS	S3VM	0.1859	0.5798	0.2343
0.1487	0.0768	0.7745	Predict ↑ True →	0.050	0.1261	0.1371	0.7368

0.8441	0.0879	0.0680	Unanno-Imbalanced (U-Im)		0.5716	0.1707	0.2577
0.1874	0.5963	0.2164	OURS	S3VM	0.1960	0.4270	0.3769
0.1321	0.0991	0.7688	Predict ↑ True →	0.025	0.1028	0.1444	0.7528

0.9126	0.0611	0.0263	Unanno-Balanced (U-Ba)		0.8055	0.0713	0.1233
0.1887	0.6722	0.1391	OURS	S3VM	0.1910	0.6049	0.2040
0.1354	0.0806	0.7840	Predict ↑ True →	0.100	0.1168	0.0916	0.7916

0.8736	0.0520	0.0745	Unanno-Balanced (U-Ba)		0.6832	0.1401	0.1767
0.1835	0.6402	0.1762	OURS	S3VM	0.1860	0.5814	0.2325
0.1224	0.0831	0.7945	Predict ↑ True →	0.050	0.1261	0.1382	0.7357

0.7562	0.1341	0.1098	Unanno-Balanced (U-Ba)		0.5718	0.1737	0.2545
0.1827	0.6564	0.1609	OURS	S3VM	0.1972	0.4297	0.3731
0.1203	0.1447	0.7350	Predict ↑ True →	0.025	0.1061	0.1474	0.7465

0.9017	0.0688	0.0295	Annotated Only (AO)		0.9107	0.0526	0.0366
0.1238	0.7465	0.1297	OURS	Super-vised DL	0.1569	0.6596	0.1835
0.0799	0.1076	0.8125	Predict ↑ True →	1.000	0.0991	0.0713	0.8296

0.8886	0.0513	0.0601	Annotated Only (AO)		0.8716	0.0711	0.0573
0.2150	0.5854	0.1995	OURS	Super-vised DL	0.2085	0.6146	0.1769
0.1432	0.0715	0.7853	Predict ↑ True →	0.100	0.1469	0.0904	0.7627

0.8013	0.0774	0.1213	Annotated Only (AO)		0.7861	0.1331	0.0808
0.2728	0.5138	0.2134	OURS	Super-vised DL	0.2283	0.5861	0.1855
0.1462	0.0991	0.7547	Predict ↑ True →	0.050	0.1611	0.1084	0.7305

0.7030	0.1567	0.1402	Annotated Only (AO)		0.5503	0.2763	0.1734
0.2229	0.5316	0.2455	OURS	Super-vised DL	0.1556	0.6051	0.2393
0.1842	0.1019	0.7139	Predict ↑ True →	0.025	0.0988	0.2242	0.6770

Figure 7. Confusion matrices (3 × 3) of the class-balanced test set. Each diagonal element of each matrix indicates the ratio of the number of correctly predicted patches in the corresponding class to the number of patches in the true class. (a) The matrices in the left column represent the semi-supervised settings (U-Im and U-Ba). (b) The matrices the right column represents the supervised settings (AO). The number next to the arrows between two matrices indicates the annotation ratio.

Table 5. Prediction accuracies and F1 scores for the class-imbalanced test set.

Class-imbal.	**Accuracy**						**F1 score (three classes, weighted averaging)**					
tests set	**Semi-supervised**				**Supervised**		**Semi-supervised**				**Supervised**	
Annotation	**U-Im**		**U-Ba**		**AO**		**U-Im**		**U-Ba**		**AO**	
ratio	**Ours**	**S3VM**	**Ours**	**S3VM**	**Ours**	**SDL**	**Ours**	**S3VM**	**Ours**	**S3VM**	**Ours**	**SDL**
1.000					0.9026	**0.9098**					0.9350	**0.9382**
0.100	**0.8621**	0.8013	**0.9099**	0.7996	**0.8809**	0.8653	**0.9095**	0.8713	**0.9392**	0.8702	**0.9202**	0.9112
0.050	**0.8617**	0.6860	**0.8676**	0.6858	**0.7944**	0.7789	**0.9088**	0.7952	**0.9121**	0.7950	**0.8672**	0.8580
0.025	**0.8412**	0.5628	**0.7498**	0.5623	**0.6988**	0.5436	**0.8969**	0.7018	**0.8390**	0.7012	**0.8044**	0.6863

S3VM (Bagattini *et al.*, 2017) and the standard SDL models are introduced as the benchmarks. The best result is highlighted in bold.

a more precise estimation of biomass or fish abundance. We will also investigate the proposed method in other domains of structured data analysis to assess whether our method generalizes to other applications. We are also interested in developing the neural networks that process missing data using internal computational mechanisms, as the missing ping is commonly found during data acquisition phase and can deteriorate the robustness of the analysis.

Supplementary Data

Supplementary material is available at the *ICESJMS* online version of the manuscript.

Data Availability Statement

The data underlying this article were provided by the Institute of Marine Research of Norway by permission. Data will be shared on request to the corresponding author with permission of the Institute of Marine Research of Norway.

Acknowledgements

This work was financially supported by the Research Council of Norway (RCN), through its project for Ubiquitous Cognitive Computer Vision for Marine Services (COGMAR, grant no. 270966) and its Center for Research-based Innovation in Marine Acoustic Abundance Estimation and Backscatter Classification (CRIMAC, grant no. 309512).

REFERENCES

Bagattini, F., Cappanera, P., and Schoen, F. 2017. Lagrangean-based combinatorial optimization for large-scale S3VMs. IEEE Transactions on Neural Networks and Learning Systems, 29: 4426–4435.

Biernacki, C., Celeux, G., and Govaert, G. 2000. Assessing a mixture model for clustering with the integrated completed likelihood. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22: 719–725.

Brautaset, O., Waldeland, A. U., Johnsen, E., Malde, K., Eikvil, L., Salberg, A.-B., and Handegard, N. O. 2020. Acoustic classification in multifrequency echosounder data using deep convolutional neural networks. ICES Journal of Marine Science. 77: 1391–1400.

Buda, M., Maki, A., and Mazurowski, M. A. 2018. A systematic study of the class imbalance problem in convolutional neural networks. Neural Networks, 106: 249–259.

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. 2018. Deep clustering for unsupervised learning of visual features. European Conference on Computer Vision (ECCV), pp. 132–149.

Chapelle, O., Scholkopf, B., and Zien, A. 2009. Semi-supervised learning. IEEE Transactions on Neural Networks, 20: 542–542.

Daan, N., Bromley, P., Hislop, J., and Nielsen, N. 1990. Ecology of north sea fish. Netherlands Journal of Sea Research, 26: 343–386.

Frederiksen, M., Furness, R. W., and Wanless, S. 2007. Regional variation in the role of bottom-up and top-down processes in controlling sandeel abundance in the north sea. Marine Ecology Progress Series, 337: 279–286.

Furness, R. W. 2002. Management implications of interactions between fisheries and sandeel-dependent seabirds and seals in the north sea. ICES Journal of Marine Science, 59: 261–269.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. 2016. Deep Learning, vol. 1. MIT Press, Cambridge.

Handegard, N. O., and Tjøstheim, D. 2009. The sampling volume of trawl and acoustics: estimating availability probabilities from observations of tracked individual fish. Canadian Journal of Fisheries and Aquatic Sciences, 66: 425–437.

ICES 2017. Report of the Benchmark Workshop on Sandeel (WKSand 2016), 31 October - 4 November 2016, Bergen, Norway. International Council for the Exploration of the Sea (ICES). ICES Document CM 2016/ACOM:33. 319 pp.

Ioffe, S. and Szegedy, C. 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. International Conference on Machine Learning (ICML), pp. 448–456.

Jabi, M., Pedersoli, M., Mitiche, A., and Ayed, I. B. 2019. Deep clustering: on the link between discriminative models and k-means. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43: 1887–1896.

Johnsen, E., Pedersen, R., and Ona, E. 2009. Size-dependent frequency response of sandeel schools. ICES Journal of Marine Science, 66: 1100–1105.

Johnsen, E., Rieucau, G., Ona, E., and Skaret, G. 2017. Collective structures anchor massive schools of lesser sandeel to the seabed, increasing vulnerability to fishery. Marine Ecology Progress Series, 573: 229–236.

Kampffmeyer, M., Løkse, S., Bianchi, F. M., Livi, L., Salberg, A.-B., and Jenssen, R. 2019. Deep divergence-based approach to clustering. Neural Networks, 113: 91–101.

Kingma, D. P. and Ba, J. 2015. Adam: a method for stochastic optimization. International Conference on Learning Representations (ICLR).

Kloser, R., Ryan, T., Sakov, P., Williams, A., and Koslow, J. 2002. Species identification in deep water using multiple acoustic frequencies. Canadian Journal of Fisheries and Aquatic Sciences, 59: 1065–1077.

Korneliussen, R. J. 2018. Acoustic target classification. ICES Cooperative Research Report No. 344. 104 pp. International Council for the Exploration of the Sea (ICES).

Korneliussen, R. J., Heggelund, Y., Macaulay, G. J., Patel, D., Johnsen, E., and Eliassen, I. K. 2016. Acoustic identification of marine species using a feature library. Methods in Oceanography, 17: 187–205.

Korneliussen, R. J., and Ona, E. 2003. Synthetic echograms generated from the relative frequency response. ICES Journal of Marine Science, 60: 636–640.

Long, J., Shelhamer, E., and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440.

MacLennan, D. N., Fernandes, P. G., and Dalen, J. 2002. A consistent approach to definitions and symbols in fisheries acoustics. ICES Journal of Marine Science, 59: 365–369.

MacLennan, D. N., and Simmonds, E. J. 2013. Fisheries Acoustics, vol. 5. Springer Science & Business Media. Berlin, Germany.

Nair, V., and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. International Conference on Machine Learning (ICML), p. 807–814.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z. *et al.* 2017. Automatic differentiation in pytorch. NIPS 2017 Workshop on Autodiff.

Prechelt, L. 1998. Early stopping-but when?In Neural Networks: Tricks of the Trade, pp. 55–69. Springer. New York City, USA.

Raitt, D. 1934. A preliminary account of the sandeels of scottish waters. ICES Journal of Marine Science, 9: 365–372.

Reid, D. G. 2000. Report on echo trace classification. ICES Cooperative Research Report No. 238. International Council for the Exploration of the Sea (ICES).

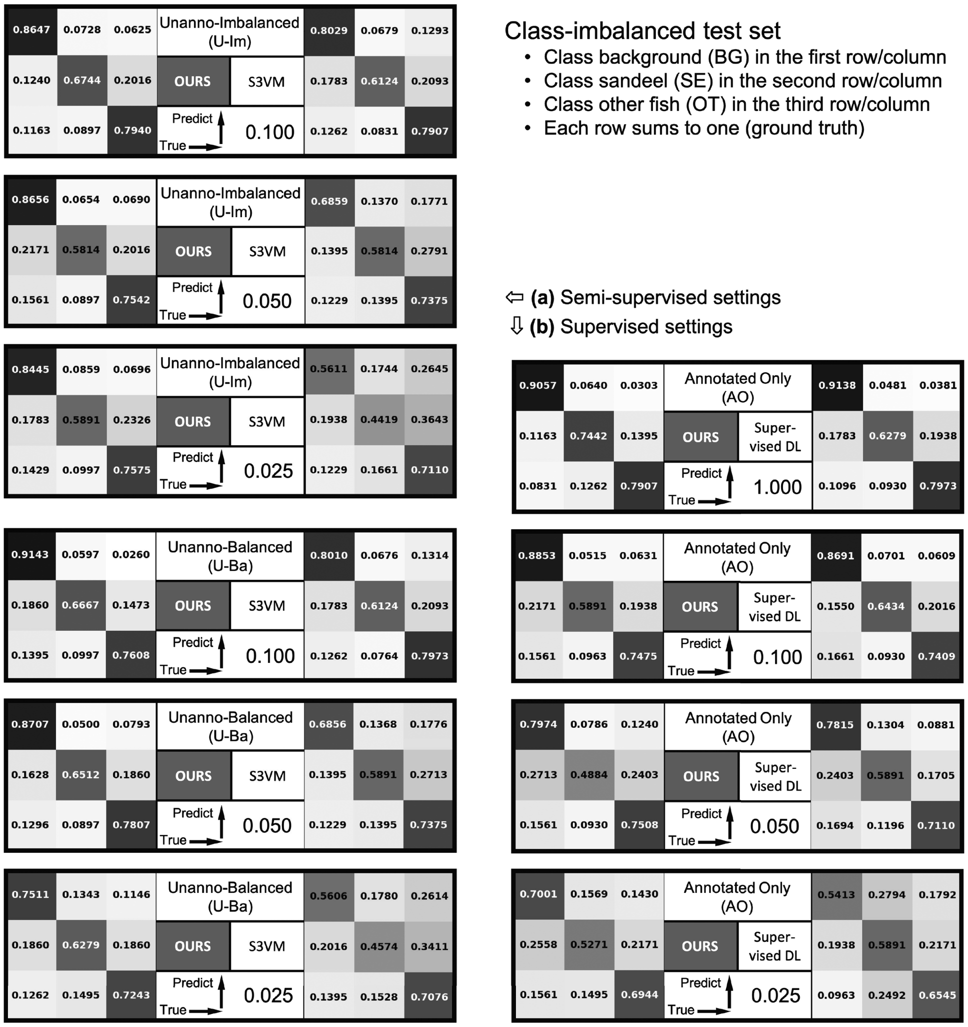
Rezvanifar, A., Marques, T. P., Cote, M., Albu, A. B., Slonimer, A., Tolhurst, T., Ersahin, K. *et al.* 2019. A deep learning-based framework for the detection of schools of herring in echograms. Tackling Climate Change with Machine Learning Workshop at Advances in Neural Information Processing Systems (NeurIPS).

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15: 1929–1958.

Van der Maaten, L., and Hinton, G. 2008. Visualizing data using t-SNE. Journal of Machine Learning Research, 9: 2579–2605.

Wold, S., Esbensen, K., and Geladi, P. 1987. Principal component analysis. Chemometrics and intelligent laboratory systems, 2: 37–52.

Yang, B., Fu, X., Sidiropoulos, N. D., and Hong, M. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. International Conference on Machine Learning (ICML), pp. 3861–3870.



Class-imbalanced test set
- Class background (BG) in the first row/column
- Class sandeel (SE) in the second row/column
- Class other fish (OT) in the third row/column
- Each row sums to one (ground truth)

⇦ **(a)** Semi-supervised settings
⇩ **(b)** Supervised settings

0.8647	0.0728	0.0625	Unanno-Imbalanced (U-Im)		0.8029	0.0679	0.1293
0.1240	0.6744	0.2016	OURS	S3VM	0.1783	0.6124	0.2093
0.1163	0.0897	0.7940	Predict ↑ True →	0.100	0.1262	0.0831	0.7907

0.8656	0.0654	0.0690	Unanno-Imbalanced (U-Im)		0.6859	0.1370	0.1771
0.2171	0.5814	0.2016	OURS	S3VM	0.1395	0.5814	0.2791
0.1561	0.0897	0.7542	Predict ↑ True →	0.050	0.1229	0.1395	0.7375

0.8445	0.0859	0.0696	Unanno-Imbalanced (U-Im)		0.5611	0.1744	0.2645
0.1783	0.5891	0.2326	OURS	S3VM	0.1938	0.4419	0.3643
0.1429	0.0997	0.7575	Predict ↑ True →	0.025	0.1229	0.1661	0.7110

0.9143	0.0597	0.0260	Unanno-Balanced (U-Ba)		0.8010	0.0676	0.1314
0.1860	0.6667	0.1473	OURS	S3VM	0.1783	0.6124	0.2093
0.1395	0.0997	0.7608	Predict ↑ True →	0.100	0.1262	0.0764	0.7973

0.8707	0.0500	0.0793	Unanno-Balanced (U-Ba)		0.6856	0.1368	0.1776
0.1628	0.6512	0.1860	OURS	S3VM	0.1395	0.5891	0.2713
0.1296	0.0897	0.7807	Predict ↑ True →	0.050	0.1229	0.1395	0.7375

0.7511	0.1343	0.1146	Unanno-Balanced (U-Ba)		0.5606	0.1780	0.2614
0.1860	0.6279	0.1860	OURS	S3VM	0.2016	0.4574	0.3411
0.1262	0.1495	0.7243	Predict ↑ True →	0.025	0.1395	0.1528	0.7076

0.9057	0.0640	0.0303	Annotated Only (AO)		0.9138	0.0481	0.0381
0.1163	0.7442	0.1395	OURS	Super-vised DL	0.1783	0.6279	0.1938
0.0831	0.1262	0.7907	Predict ↑ True →	1.000	0.1096	0.0930	0.7973

0.8853	0.0515	0.0631	Annotated Only (AO)		0.8691	0.0701	0.0609
0.2171	0.5891	0.1938	OURS	Super-vised DL	0.1550	0.6434	0.2016
0.1561	0.0963	0.7475	Predict ↑ True →	0.100	0.1661	0.0930	0.7409

0.7974	0.0786	0.1240	Annotated Only (AO)		0.7815	0.1304	0.0881
0.2713	0.4884	0.2403	OURS	Super-vised DL	0.2403	0.5891	0.1705
0.1561	0.0930	0.7508	Predict ↑ True →	0.050	0.1694	0.1196	0.7110

0.7001	0.1569	0.1430	Annotated Only (AO)		0.5413	0.2794	0.1792
0.2558	0.5271	0.2171	OURS	Super-vised DL	0.1938	0.5891	0.2171
0.1561	0.1495	0.6944	Predict ↑ True →	0.025	0.0963	0.2492	0.6545

Figure A1. Confusion matrices (3 × 3) of the class-imbalanced test set, where the first row/column indicates BG, the second one is OT, and the third one is SE.

Table A1. The number of test patches sampled in a way to preserve the intrinsic class imbalance from the test echosounder data (2018–2019).

Year	**Test set (2018–2019)**	
Class	**Extracted patches (percentage)**	**Sampled by intrinsic distr.**
BG	816 726 (97.61)	17 582
OT	6004 (0.72)	129
SE	13 984 (1.67)	301
Total	836 714 (100.00)	18 012

Appendix

Deep learning terminologies

Epoch indicates that the model has performed a single pass over the entire training set.

Loss function is a measure of how good a model is performing for a specific task. A high value of the loss function indicates poor model performance. In order to improve the performance of the model for the given task, the loss is minimized.

One-hot encoding is a method to quantify categorical data by producing a vector with length equal to the number of categories in the data set. If a data point belongs to the i^{th} category then all elements of this vector are assigned the value 0 except for the i^{th} component which is assigned a value of 1.

Softmax function is a generalization of the logistic function to multiple dimensions. It is used in multi-class classification and is often used as the last activation function of a neural network to normalize the output of a network to a probability distribution over predicted output classes.

Cross-entropy loss measures the performance of a classification model whose output is a probability value between 0 and 1. The cross-entropy loss increases as the predicted probability diverges from the actual label. The ideal model would have the loss of 0, where an outcome of the model has a form of a one-hot vector.

End-to-end learning model refers to training a possibly complex learning system represented by a single model that represents the complete target system, bypassing the intermediate layers usually present in traditional pipeline designs.

Handling Editor: Cigdem Beyan