



Article

SSL-LRN: A Lightweight Semi-Supervised-Learning-Based Approach for UWA Modulation Recognition

Chaojin Ding , Wei Su *, Zehong Xu, Daqing Gao and En Cheng

School of Information Technology, Xiamen University, Xiamen 361104, China; dingcj@stu.xmu.edu.cn (C.D.); 23320211154257@stu.xmu.edu.cn (Z.X.); dqgao@stu.xmu.edu.cn (D.G.); chengen@xmu.edu.cn (E.C.)

* Correspondence: suweixiamen@xmu.edu.cn

Abstract: Due to the lack of sufficient valid labeled data and severe channel fading, the recognition of various underwater acoustic (UWA) communication modulation types still faces significant challenges. In this paper, we propose a lightweight UWA communication type recognition network based on semi-supervised learning, named the SSL-LRN. In the SSL-LRN, a mean teacher–student mechanism is developed to improve learning performance by averaging the weights of multiple models, thereby improving recognition accuracy for insufficiently labeled data. The SSL-LRN employs techniques such as quantization and small convolutional kernels to reduce floating-point operations (FLOPs), enabling its deployment on underwater mobile nodes. To mitigate the performance loss caused by quantization, the SSL-LRN adopts a channel expansion module to optimize the neuron distribution. It also employs an attention mechanism to enhance the recognition robustness for frequency-selective-fading channels. Pool and lake experiments demonstrate that the framework effectively recognizes most modulation types, achieving a more than 5% increase in recognition accuracy at a 0 dB signal-to-noise ratio (SNRs) while reducing FLOPs by 84.9% compared with baseline algorithms. Even with only 10% labeled data, the performance of the SSL-LRN approaches that of the fully supervised LRN algorithm.

Keywords: underwater acoustic communication; modulation recognition; semi-supervised learning; lightweight convolutional neural network



Citation: Ding, C.; Su, W.; Xu, Z.; Gao, D.; Cheng, E. SSL-LRN: A Lightweight Semi-Supervised-Learning-Based Approach for UWA Modulation Recognition. *J. Mar. Sci. Eng.* **2024**, *12*, 1317. <https://doi.org/10.3390/jmse12081317>

Academic Editor: Marco Cococcioni

Received: 8 July 2024

Revised: 29 July 2024

Accepted: 31 July 2024

Published: 4 August 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Underwater acoustic (UWA) communication is the primary means of information exchange among underwater devices and plays a critical role in the detection and identification of underwater targets. Also, it is crucial for enhancing the efficiency of underwater communication networks and improving the anti-interference capabilities of intelligent UWA communication systems [1,2]. Traditional UWA communication modulation recognition schemes combine expert feature extraction with basic machine learning classification techniques and are effective at recognizing specific modulation types [3,4]. For instance, cyclostationary analysis combined with higher-order statistics is utilized to form composite features, and a support vector machine (SVM) is employed to effectively recognize multiple phase-shift keying (MPSK) and multiple frequency-shift keying (MFSK) modulation types [5]. However, these traditional methods often require accurate prior knowledge about the signal characteristics and the environment, such as the channel impulse responses (CIRs), which are difficult to estimate with known training sequences. Furthermore, the limited bandwidth available for UWA communications becomes increasingly congested with signals from different modulation schemes, thereby reducing the network throughput of the UWA sensor network. Due to the spatial and temporal variability of the UWA channel, multipath effects, and fading, UWA communication signals experience significant frequency-selective fading. Under such conditions, the recognition performance of traditional schemes based on the maximum likelihood criterion or expert features decreases

at low signal-to-noise ratios (SNRs). This decline is particularly evident with modulation types that are easily confused, such as QPSK and 8PSK [6].

Deep learning, leveraging its strengths in extracting high-dimensional features and capturing non-linear characteristics, has demonstrated improved recognition accuracy in fading channels such as UWA communication systems under low-SNR conditions [7–10]. For instance, Zhang et al. [7] proposed a hybrid R&CNN algorithm that combines convolutional neural networks (CNNs) and recurrent neural networks (RNNs) for recognizing UWA modulation signals. Sea trial results indicate that this method significantly improves recognition accuracy under low-SNR conditions. In addition to improvements in network structure, some researchers have focused on signal preprocessing and denoising, achieving significant results. Yao et al. [8] proposed a deep complex matched filter and a deep complex channel equalizer based on neural networks to denoise received signals and reduce the impact of multipath fading, thereby effectively enhancing recognition accuracy. Wang et al. [9] introduced the IAFNET algorithm, which incorporates a noise elimination module and a task-driven mechanism to extract features more effectively. Furthermore, to optimize the separation of feature spaces, Gao et al. [10] proposed the UMCSCSCL algorithm, which employs supervised contrastive learning, further improving recognition performance.

However, these algorithms cannot be deployed on underwater devices such as buoys and unmanned underwater vehicles (UUVs), which have limited computational capability and energy. The floating-point operations (FLOPs) of neural networks are a critical indicator that determines whether deep learning algorithms can be successfully deployed on devices with limited computational power and energy, such as underwater nodes. In recent years, significant progress has been made in lightweight modulation recognition research. For instance, input data can be transformed into different coordinate systems to highlight significant features. Additionally, feature maps can be folded to reduce redundant information. These techniques reduce complexity while enhancing the neural network's ability to extract features [11]. In the area of signal processing, Bai et al. [12] proposed an adaptive denoising network based on convolution, which effectively filters noise in signals and reduces computational complexity during network training. Additionally, to improve network efficiency, Lyu et al. [13] introduced depthwise separable convolutions and convolutional attention modules in ResNet18, which reduces network size while improving acoustic signal recognition performance. In terms of feature extraction, the SSKNET algorithm is designed with special convolutional kernels to extract deeper semantic features with fewer layers [14]. However, the performance of these lightweight algorithms may decline compared to full-precision algorithms.

Deep learning models typically require a large number of high-quality training samples [15]. However, acquiring sufficient labeled data for UWA modulation recognition tasks is challenging and costly [16]. To effectively utilize unlabeled data, Lee et al. [17] introduced pseudo-label learning, which demonstrated excellent performance on the MNIST dataset. To further improve the utilization of unlabeled data, many semi-supervised learning methods incorporate loss terms for unlabeled data. For instance, Dong et al. [18] proposed the SSRCNN algorithm, which employs random perturbations of unlabeled signals and is designed with a divergence-based loss function for unlabeled data, effectively extracting information from these signals. Liu et al. [19] proposed the SEMIAMC algorithm, which uses self-supervised contrastive learning to pre-train with unlabeled data, achieving higher performance with limited labeled data. To further enhance feature extraction, Guo et al. [20] used a short-time Fourier transform (STFT) to extract modulation features and enhanced the model's generalization ability through pseudo-labels, consistency constraints, and entropy regularization, significantly improving modulation recognition accuracy with minimal labeled data. Fu et al. [21] proposed the SS-SEI algorithm, which is based on metric adversarial training and uses a regularized objective function to extract semantic features, thereby achieving better recognition performance. However, these algorithms are not specifically designed for UWA channels.

In this paper, we propose a lightweight UWA modulation recognition algorithm, lightweight recognition network (LRN), designed to identify common acoustic modulation types such as MFSK, MPSK, multiple quadrature amplitude modulation (MQAM), and orthogonal frequency division multiplexing (OFDM). Traditional deep learning algorithms typically use large convolutional kernels and have high computational complexity, making them difficult to deploy on underwater nodes. The LRN algorithm uses numerous small kernels instead of traditional large ones and uses quantized weights and parameters to reduce the FLOPs, making it more suitable for underwater devices. A channel expansion module is introduced to optimize the neuron distribution within the network, enhancing feature learning capabilities and improving recognition accuracy. Additionally, an attention mechanism is established to accelerate the training speed and enhance robustness. Deep learning models generally rely on large amounts of high-quality training samples, but obtaining such samples in UWA channels is very challenging. To effectively utilize unlabeled data, we also propose a semi-supervised-learning LRN (SSL-LRN) algorithm. In the SSL-LRN algorithm, interpolation consistency training is applied to unlabeled data; this is combined with a mean teacher–student mechanism to enhance the stability of pseudo-labels, thereby improving the efficiency of extracting useful features from unlabeled data. We have established a UWA modulation recognition dataset that includes nine modulation types to evaluate performance; the dataset features simulated data as well as pool and lake test data. The performance of the SSL-LRN algorithm is analyzed and compared with two other semi-supervised learning-based algorithms and three newly proposed recognition algorithms.

The principal contributions of this paper are outlined as follows:

1. We propose the LRN for UWA communication modulation recognition, which utilizes a streamlined network architecture with small convolutional kernels and incorporates a channel expansion module and an attention mechanism. This algorithm significantly reduces FLOPs while achieving high recognition accuracy for various modulation types such as MFSK, MPSK, MQAM, and OFDM under low-SNR conditions.
2. We also introduce an SSL-LRN algorithm for acoustic communication modulation recognition. This algorithm uses interpolation consistency training for unlabeled data and incorporates an average teacher model to stabilize pseudo-labels. This approach improves learning efficiency from unlabeled data, enhances robustness, and boosts learning capabilities in scenarios with insufficient labeled data [9,10].
3. The performance of the LRN algorithm is compared to that of three baseline algorithms. The LRN algorithm exhibits the lowest FLOPs: reduced by more than 84.9%. In pool and lake experiments, the LRN algorithm demonstrates the highest recognition accuracy at 0 dB, reaching 95.5%.

The structure of the paper is organized as follows: In Section 2, we introduce the system model and signal preprocessing methods. In Section 3, the architecture of the proposed semi-supervised lightweight network is detailed. In Section 4, we present experimental results conducted in pool and lake environments and compare these with baseline algorithms. In Section 5, conclusions are drawn.

2. Signal Preprocessing

2.1. System Model

As shown in Figure 1, the underwater data collector is equipped with an omnidirectional hydrophone and a UWA communication system that broadcasts communication signals $s(t)$ to communicate with buoys [7]. Both buoys and UUVs can receive these signals using hydrophones, which typically consist of synchronization and modulation components. The buoy has comprehensive knowledge of the synchronization signal structure, modulation type, and code length, which it uses to demodulate $x(t)$ and extract necessary information. UUVs also aim to derive useful information from $x(t)$, but they must first identify the modulation type and extract features using embedded devices [22].

However, the accuracy of such tasks performed by embedded devices is limited by their computational power, energy consumption, and algorithm efficiency.

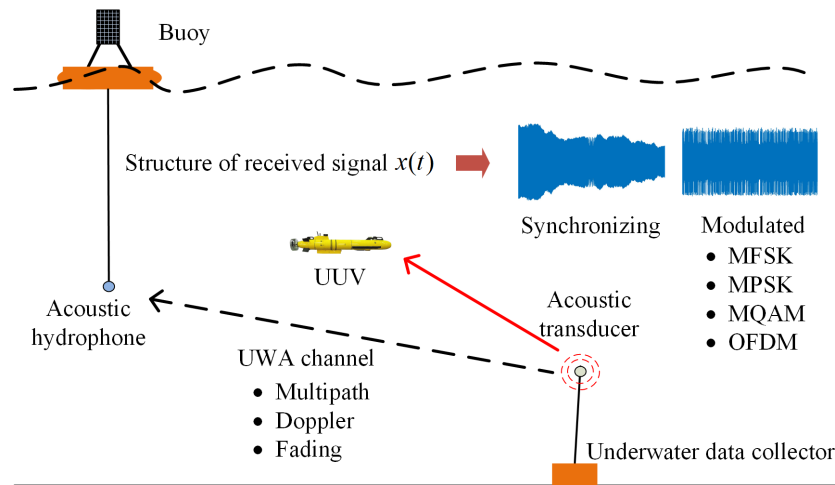


Figure 1. System model.

In our calculations, we used the double datatype (float64) to ensure higher precision in our results. This datatype was chosen to minimize the impact of numerical errors and to enhance the robustness of the signal processing and recognition algorithms.

Through the UWA channel, the transmitted signal $s(t)$ is affected by multipath effects, fading, and Doppler shifts. According to the ray acoustic model, the received signal $x(t)$ can be expressed as:

$$x(t) = \sum_{i=1}^N A_i s(\alpha_i(t - \tau_i)) + n(t) \tag{1}$$

where $x(t)$ represents the received signal, N is the total number of paths, A_i denotes the attenuation amplitude of the i -th path, α_i is the Doppler scale factor for the i -th path, τ_i is the time delay associated with the i -th path, and $n(t)$ is the additive noise.

Due to the complexity of UWA channels and the limited computational capacity of embedded devices, it is crucial to preprocess the received signals to enhance the accuracy and robustness of modulation type recognition. Signal preprocessing helps convert raw time-domain signals into a data format that the recognition algorithm can efficiently process.

2.2. Signal Preprocessing

It is important to note that the original time-domain signal contains the richest information. To discern the differences between adjacent symbols and enhance the accuracy and robustness of recognition, the recognition algorithm requires the input of several symbols. However, the symbol durations for some UWA modulation types, such as MFSK and OFDM, can extend beyond tens of milliseconds. In such cases, using the original time-domain signals directly as inputs becomes overly complex. As described in [23], by judiciously designing the structure of the recognition network, including activation functions and loss functions, similar recognition accuracy can be achieved with preprocessed signals. The received signals are preprocessed as follows:

2.2.1. Segmentation and Normalization

The received signal is segmented into smaller sections and filtered using a band-pass filter. The normalization of these segments is defined as:

$$\tilde{x}(n) = \frac{Nx(n)}{\sum_{i=1}^N x(n)} \tag{2}$$

where $x(n)$ represents the sampled signal and N is the number of samples.

2.2.2. Transforming and Mapping

Inspired by [23], we apply the STFT to the signal to obtain its complex representation in the time–frequency domain. The STFT provides a localized frequency representation of the signal, which is crucial for analyzing non-stationary signals like UWA signals:

$$X(m, k) = \sum_{n=0}^{M-1} \tilde{x}(n)w(n - m)e^{-j2\pi kn/M} \tag{3}$$

where $w(n)$ is the Hann window function. The Hann window is utilized to reduce discontinuities at the signal ends, which helps minimize spectral leakage and edge effects that can distort the time–frequency representation. The function is defined as:

$$w(n) = \begin{cases} 0.5[1 - \cos(\frac{2\pi n}{M-1})] & 0 \leq n \leq M - 1 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Subsequently, we calculate the magnitude and phase of $X(m, k)$:

$$|X(m, k)| = \sqrt{\Re(X(m, k))^2 + \Im(X(m, k))^2} \tag{5}$$

$$\angle X(m, k) = \arctan 2(\Im(X(m, k)), \Re(X(m, k))) \tag{6}$$

Here, $\Re(\cdot)$ and $\Im(\cdot)$ denote the real and imaginary parts of the complex numbers, respectively. The magnitude $|X(m, k)|$ reflects the energy intensity across different frequency components, which is crucial for capturing energy patterns and variations. The phase $\angle X(m, k)$ provides relative timing information between frequency components, which is essential for recognizing phase-sensitive modulation types such as MPSK signals.

By combining $|X(m, k)|$ and $\angle X(m, k)$, we obtain a comprehensive representation of the signal in the time–frequency domain. Mapping these two types of information onto two 64×64 matrices, we create a dual-channel input for the recognition network, where one channel carries amplitude information and the other carries phase information. This data structure allows the network to fully utilize the signal’s energy distribution characteristics and waveform information, thus enhancing the accuracy of modulation type recognition.

3. The Proposed Recognition Algorithm

3.1. The Structure of the Proposed LRN Algorithm

The structure of the LRN is illustrated in Figure 2. The architecture comprises one layer of conventional convolution (Conv), five layers of quantized convolution (QConv), three layers of max pooling (Maxpool), and one fully connected (FC) layer. Inspired by the VGG16 network architecture, the LRN algorithm does not utilize large convolutional kernels but instead employs multiple small kernels [24], effectively reducing computational complexity. In the diagram, light brown blocks represent the full-precision Conv layers, dark brown blocks denote the quantized QConv layers, light blue blocks represent the Maxpool layers, and yellow blocks signify the FC layer. Maxpool2 indicates a kernel size of 2×2 for the max pooling layer, and Conv3 indicates a kernel size of 3×3 for the convolution layer. The symbols s and p respectively denote stride and padding.

When deploying neural networks on embedded devices, the number of FLOPs serves as a critical metric to gauge the computational complexity of the network. It reflects the computational resources required for one forward pass of the network [25,26]. Although the LRN algorithm has fewer convolutional layers compared to VGG16, its FLOPs may still be relatively high due to the number of parameters and the computational intensity involved. High FLOPs mean that the network requires more computations per input, making it challenging to deploy on resource-constrained embedded devices.

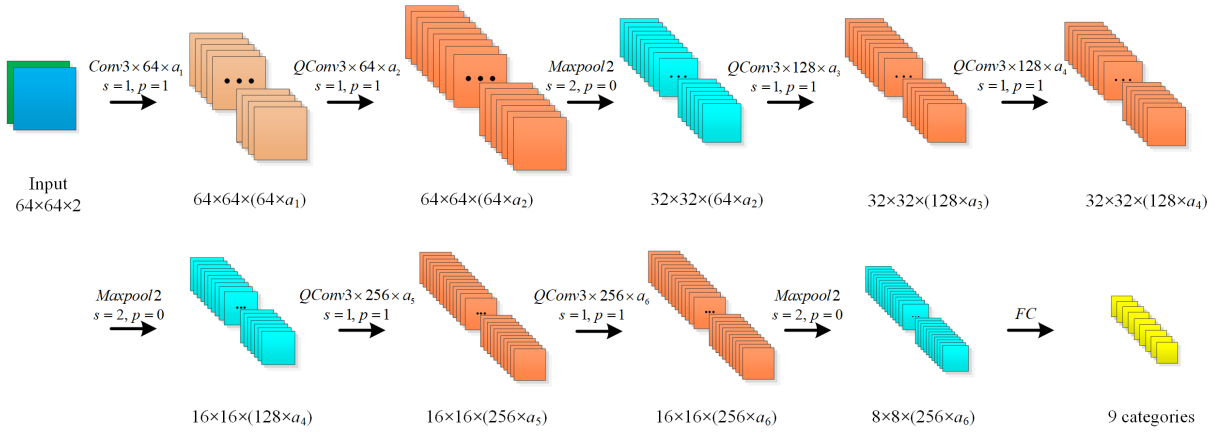


Figure 2. The structure of the proposed LRN algorithm.

To address the issue of high computational demands, inspired by the binary neural network approach proposed in [25], we optimize the LRN by using only binary weights and parameters to significantly reduce the number of FLOPs. The quantization of floating-point weights W and activation parameters A is performed as follows:

$$\begin{cases} W_b = \text{sign}(W) \times E(|W|) \\ A_b = \text{round}(\text{Clip}(A, 0, 1)) \\ \quad = \text{round}(\max(0, \min(1, A))) \end{cases} \quad (7)$$

where W_b and A_b are the binary forms of W and A , respectively. The function $\text{sign}(W)$ represents the sign function. $E(|\cdot|)$ calculates the expected value of the absolute values. The function $\text{Clip}(A, 0, 1)$ restricts the values of A to between 0 and 1, ensuring that the activation parameters remain within a valid range.

Due to the first convolutional layer interacting directly with the input image and containing a wealth of feature information, quantization at this stage can result in a substantial loss of critical information. Similarly, the last fully connected layer, which contains highly refined information after multiple layers of convolution and pooling, is also not suitable for quantization. Therefore, the initial convolution layer and the final output layer of the network are not subjected to quantization. This approach preserves the accuracy of important information and features. Meanwhile, the weights and activations of the remaining layers are quantized. As shown in Figure 2, except for the first convolutional layer and the fully connected layer, all convolutional layers restrict the weights and activation parameters to 1 bit. For example, QConv3 indicates the quantization of weights for the kernel.

The input of dimensions $64 \times 64 \times 2$ is initially processed by the first Conv layer and the second QConv layer. After each convolution operation, an activation function extracts features, which are then refined by a max pooling layer to remove redundant information, thereby enhancing processing efficiency and recognition accuracy. As illustrated in Figure 2, the features are processed through a series of convolution and pooling layers, and ultimately, the recognition results are output by the FC layer.

3.2. Performance Improvement of the LRN Algorithm

The reduction in the number of convolution layers and the quantization of parameters decrease the FLOPs of the LRN algorithm. However, the quantization process can introduce new challenges such as gradient loss during backpropagation, slower training speeds, and decreased recognition accuracy.

3.2.1. The Training Algorithm of the LRN

Initially, due to the quantization of parameters and weights, gradient vanishing occurs during backpropagation when partial derivatives of the loss function with respect

to the activation values are calculated, preventing the network from updating. This is illustrated by:

$$\frac{\partial L_{oss}}{\partial A} = \frac{\partial L_{oss}}{\partial A_b} \times \frac{\partial A_b}{\partial A} = \frac{\partial L_{oss}}{\partial A_b} \times 0 = 0 \tag{8}$$

where L_{oss} denotes the loss function. To address this issue, a straight-through estimator is constructed according to [27], where the derivative $\partial A_b / \partial A$ is replaced by the derivative of a hard \tanh function, which is represented as:

$$\frac{dHtanh(A)}{dA} = \frac{dClip(A, -1, 1)}{dA} = 1_{|A| \leq 1} \tag{9}$$

Secondly, to enhance the network’s nonlinearity, activation functions are introduced after each convolutional and fully connected layer. Typically, the $ReLU$ function is employed due to its fast convergence and computational efficiency. However, this can result in the loss of negative value information. To address this issue, ref. [28] proposed the $LeakyReLU$ function, which replaces the negative values of the $ReLU$ function with leaky values, thereby preserving negative values. This is shown by:

$$LeakyReLU(x) = \max(kx, x) = \begin{cases} x & x \geq 0 \\ kx & x < 0 \end{cases} \tag{10}$$

where $k \in \{0, 1\}$.

Finally, to further accelerate training, a batch normalization step is incorporated before the activation functions; it normalizes the data distribution to have a mean of zero and a variance of one. According to [29], the batch normalization operation is defined as:

$$y = \frac{\eta - \frac{1}{p} \sum_{i=1}^p \eta^{(i)}}{\text{sqrt}\left(\frac{1}{p} \sum_{i=1}^p \left(\eta^{(i)} - \frac{1}{p} \sum_{i=1}^p \eta^{(i)}\right)^2\right)} \tag{11}$$

where η represents the feature point and p is the total number of feature points.

The training algorithm of the LRN is outlined in Algorithm 1. Additionally, we integrate a channel expansion module and an attention mechanism, which significantly enhance recognition accuracy with only a slight increase in FLOPs, parameter count, and memory usage.

Algorithm 1 LRN Training Algorithm.

Require: Hyperparameter learning rate α and labeled input and target set $\{(X_1, Y_1), \dots, (X_n, Y_n)\}$.

- 1: **Initialization:** Initialize weights W and biases B for all layers.
 - 2: L_{oss} is the cost function, W is weights, B is bias, and L is the number of layers.
 - 3: **{1.Forward propagation}**
 - 4: **for** $k = 1$ to L **do**
 - 5: Binarize the weights and biases:
 - 6: $W_b^k = \text{sign}(W^k) * E(|W^k|)$
 - 7: $B_b^k = \text{sign}(B^k) * E(|B^k|)$
 - 8: Compute the linear transformation:
 - 9: $Z^k = \text{sign}(W_b^k) * A_b^{k-1} + B_b^k$
 - 10: Apply activation function ($LeakyReLU$):
 - 11: $A^k = LeakyReLU(Z^k)$
-

Algorithm 1 Cont.

```

12:   if  $k < L$  then
13:       Clip and binarize the activations:
14:        $A_b^k = \text{round}(\text{Clip}(A^k, 0, 1))$ 
15:   end if
16: end for
17: {2.Backward propagation}
18: Compute the gradient of the loss function with respect to the final activation:
19: Compute  $g_{A_L} = \frac{\partial L_{\text{loss}}}{\partial A_L}$ , knowing  $A$  and  $Y$ 
20: for  $k = L$  to 1 by  $-1$  do
21:   if  $k < L$  then
22:     Compute the gradient of the activation function:
23:      $g_{A^k} = g_{A_b^k} * \frac{d\text{Htanh}(A^k)}{dA^k} = g_{A^k}^b * 1_{|A^k| \leq 1}$ 
24:   end if
25:   Compute the gradient with respect to  $Z^k$ :
26:    $g_{Z^k} = g_{A^k} * \frac{dA^k}{dZ^k}$ 
27:   Propagate the gradient to the previous layer:
28:    $g_{A^{k-1}}^b = W_b^k * g_{Z^k}$ 
29:   Compute the gradients with respect to weights and biases:
30:    $g_{W_b^k} = A_b^{k-1} * g_{Z^k}$ 
31:    $g_{B_b^k} = \text{sum}(g_{Z^k})$ 
32:   Update the weights and biases:
33:    $W_b^k = W_b^k - \alpha g_{W_b^k}$ 
34:    $B_b^k = B_b^k - \alpha g_{B_b^k}$ 
35: end for

```

3.2.2. Channel Expansion Modules

Following quantization, there may be a decline in recognition accuracy due to the loss of neuronal precision. According to [26], the accuracy of recognition is influenced by both the number and the precision of neurons. To enhance recognition accuracy, we introduce channel expansion modules. As shown in Figure 2, channel expansion is achieved by increasing the number of channels in specific layers, optimizing neuron distribution within the network, and enhancing feature learning capabilities. Specifically, a_n represents the channel expansion ratio for the n_{th} convolutional layer. Since the weights and parameters of the convolutional layers are already quantized, the inclusion of channel expansion modules only results in a slight increase in FLOPs. However, expansion also implies an increase in the number of parameters and memory usage, which will be analyzed in Section 4.

Among the many parameters that affect recognition performance, the channel expansion ratio a_n is a critical factor. According to [26], the expansion ratio for each layer is selected from the set $\{0.25, 0.5, 1, 2, 3, 4\}$. For the six convolutional layers, the channel expansion ratios are represented as a vector $a_l = [a_1, \dots, a_6]$. The challenge lies in selecting the optimal a_l from a large number of possible combinations.

To address this, we evaluated each possible a_l by calculating its recognition accuracy ε_m and FLOPs β_m . We then established an evaluation metric e_m to measure the utility of different a_l configurations, which is defined as follows:

$$e_m = \varepsilon_m - 0.1 \times \beta_m \tag{12}$$

We selected a_l by maximizing the value of e_m , ultimately determining the optimal channel expansion ratios to be $[1, 2, 1, 1, 1, 0.5]$.

3.2.3. Attention Mechanism

Inspired by [30], we incorporated an attention mechanism to further enhance the performance of the LRN. Compared to adding additional convolutional layers or parameters, the attention mechanism improves the network’s capability to extract key information with relatively low computational cost. The attention mechanism consists of two modules: the channel attention module and the spatial attention module, as depicted in Figure 3 and Figure 4, respectively.

The specific composition of the channel attention module is illustrated in Figure 3, where H , W , and C respectively represent the height, width, and number of channels of the input features. In this module, average pooling and max pooling operate concurrently to aggregate spatial information from the input feature \mathbf{F} . Following aggregation, the size of \mathbf{F} is reduced from $H \times W \times C$ to $1 \times 1 \times C$. Subsequently, a shared deep neural network (DNN) composed of three fully connected layers generates two attention maps of size $1 \times 1 \times C$. These attention maps are then superimposed and processed through a sigmoid function to produce a normalized channel attention map, which is given by:

$$A_C(\mathbf{F}) = \sigma(DNN(P_{avg}(\mathbf{F})) + DNN(P_{max}(\mathbf{F}))) \tag{13}$$

where $\sigma(\cdot)$ denotes the sigmoid function, and P_{avg} and P_{max} respectively represent average pooling and *Maxpool*.

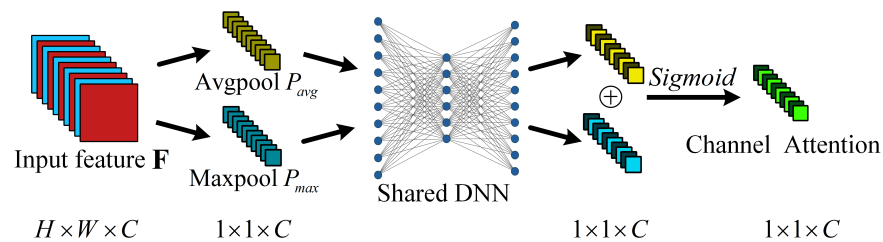


Figure 3. Channel attention module.

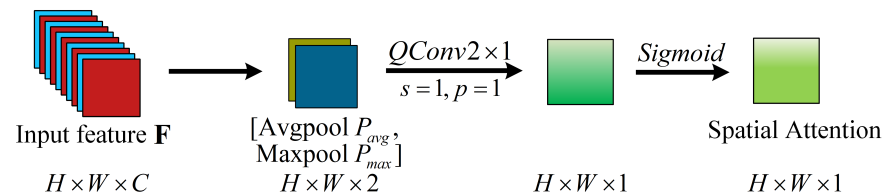


Figure 4. Spatial attention module.

Similarly, the computation process for spatial attention is illustrated in Figure 4. Initially, the channel information of \mathbf{F} is aggregated through sequential average pooling and max pooling operations, reducing the dimensions from $H \times W \times C$ to $H \times W \times 1$. The results of these pooling operations are then concatenated to form a feature map of dimensions $H \times W \times 2$. This feature map is processed through a 2×2 convolutional layer, and the output is subsequently passed through a sigmoid function to produce the final spatial attention map of dimensions $H \times W \times 1$. The normalized attention map created by the spatial attention module is represented as:

$$A_S(\mathbf{F}) = \sigma(C^{2 \times 2}([P_{avg}(\mathbf{F}); P_{max}(\mathbf{F})])) \tag{14}$$

where $C^{2 \times 2}$ denotes a convolution operation with a kernel size of 2×2 .

It is important to emphasize that the DNN module is shared and the convolution kernel weights have been quantized; thus, the increase in computational complexity is minimal. The input features are first weighted by the channel attention module to enhance the response of important channel features and suppress less significant ones. Subsequently,

the spatial attention map is multiplied element-wise with the original features at the spatial level, enhancing the focus on task-relevant areas. Through attention mechanisms in both the channel and spatial dimensions, the network more effectively captures and emphasizes critical information.

3.3. Semi-Supervised-Learning-Based LRN: SSL-LRN Algorithm

In the field of UWA communication, obtaining sufficient labeled data is not only time-consuming but also costly. Additionally, the harsh conditions of UWA channels make the labeling process prone to errors, which is a factor that can potentially degrade the performance of the LRN model. Semi-supervised learning addresses the lack of sufficient labeled data. By integrating a large amount of unlabeled data with a small amount of labeled data, this approach effectively trains neural networks. It also mitigates the effects of insufficiently labeled data [16]. The structure of the proposed SSL-LRN algorithm introduced in this study is illustrated in Figure 5. The SSL-LRN algorithm consists of two components: a student model and a teacher model (LRN-T). The student model is a standard LRN model, while the teacher model is an auxiliary network derived from the student model. The training process of the SSL-LRN algorithm is also divided into two parts.

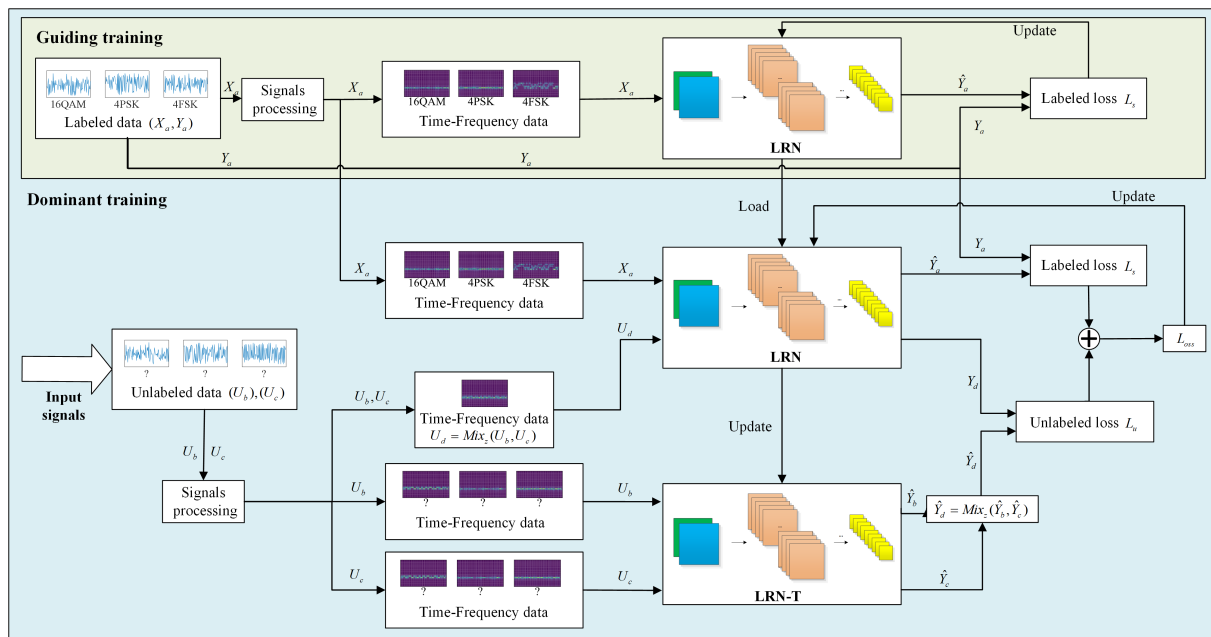


Figure 5. The flowchart of the SSL-LRN algorithm.

3.3.1. Guiding Training with Insufficient Labeled Data

During the initial training phase, the SSL-LRN model relies on a small amount of labeled data to guide the training process, ensuring the model learns in the correct direction. During this phase, cross-entropy is used as the loss function to enhance convergence speed. The loss function for the labeled data is represented by L_s , which is given by:

$$L_s = -\frac{1}{N} \sum_{a=1}^N \sum_{c=1}^M Y_{ac} \log(\hat{Y}_{ac}) \quad (15)$$

where N is the number of data points, M is the number of label categories, and \hat{Y}_{ac} is the predicted probability. If the actual class of data point a is c , then $Y_{ac} = 1$. Otherwise, $Y_{ac} = 0$.

3.3.2. Dominant Training with Unlabeled Data

Once the model demonstrates initial effectiveness, to prevent overfitting, a large volume of unlabeled data is introduced to dominate the training process. As iterations progress, the weight of these data in parameter updates gradually increases. Initially, according to [31], unlabeled data U_d and pseudo-labels Y_d are generated through a linear interpolation algorithm.

Specifically, LRN-T is constructed to assist the LRN; it features a similar network structure but has a slower rate of parameter updates. In this study, the parameters of LRN-T are designed to be the moving average of the LRN parameters, which enhances training efficiency and robustness [32]. The unlabeled data are then split into two parts: U_b and U_c . LRN-T processes U_b and U_c to compute the predicted probabilities \hat{Y}_b and \hat{Y}_c . Subsequently, pseudo data U_d and pseudo labels Y_d are created by

$$\begin{cases} U_d = \text{Mix}_z(U_b, U_c) = z \cdot U_b + (1 - z) \cdot U_c \\ Y_d = \text{Mix}_z(\hat{Y}_b, \hat{Y}_c) = z \cdot \hat{Y}_b + (1 - z) \cdot \hat{Y}_c \end{cases} \quad (16)$$

where z is a random number calculated from $\text{Beta}(b, b)$ [33]. In this study, b is set to 0.5, and z is re-estimated at each interpolation.

The loss values for LRN and LRN-T are represented by Y_d and \hat{Y}_d , respectively. The unlabeled loss L_u is calculated using the mean squared error to enhance robustness, as shown in the following equation:

$$L_u = \frac{1}{N} \sum_{i=1}^N (Y_d - \hat{Y}_d)^2 \quad (17)$$

The total loss function L_{oss} is the weighted sum of L_s and L_u . Given that the weight of the unlabeled data should increase over time, L_{oss} is constructed as:

$$L_{oss} = L_s + r(t) \cdot L_u \quad (18)$$

where $r(t)$ represents the importance of the unlabeled data. As unlabeled data are typically unstable at the start of training, $r(t)$ is a monotonically increasing function over time.

The SSL-LRN algorithm is described in Algorithm 2.

Algorithm 2 SSL-LRN Algorithm

Input: Unlabeled input set $\{U_1, U_2, \dots, U_n\}$, labeled input and target set $\{(X_1, Y_1), \dots, (X_n, Y_n)\}$, hyperparameter learning rate α , rate of moving average γ , batch size S , and ramp function $r(t)$.

Output: Parameters θ .

- 1: Initialize parameters θ .
 - 2: M is the number of label categories.
 - 3: **for** $t = 1$ to T **do**
 - 4: Feed in a labeled batch $\{(X_1, Y_1), \dots, (X_S, Y_S)\}$ to LRN to obtain \hat{Y}_a .
 - 5: Calculate supervised loss L_s :
 - 6: $L_s = -\frac{1}{S} \sum_{a=1}^S \sum_{c=1}^M Y_{ac} \log(\hat{Y}_{ac})$.
 - 7: Feed in two unlabeled batches $\{U_1, U_2, \dots, U_S\}$ and $\{U_{S+1}, U_{S+2}, \dots, U_{2S}\}$ to LRN-T to obtain \hat{Y}_b and \hat{Y}_c .
 - 8: Sample $z \sim \text{Beta}(b, b)$.
 - 9: Calculate mixed input U_d :
 - 10: $U_d = z \cdot U_b + (1 - z) \cdot U_c$.
 - 11: Calculate mixed target Y_d :
 - 12: $Y_d = z \cdot \hat{Y}_b + (1 - z) \cdot \hat{Y}_c$.
 - 13: Feed in U_d to LRN to obtain \hat{Y}_d .
 - 14: Calculate unsupervised loss L_u :
 - 15: $L_u = \frac{1}{S} \sum_{i=1}^S (Y_d - \hat{Y}_d)^2$.
-

Algorithm 2 Cont.

- 16: Calculate total loss L_{oss} :
- 17: $L_{oss} = L_s + r(t) * L_u$.
- 18: Update the LRN-T parameters:
- 19: $\theta_T = \gamma\theta_T + (1 - \gamma)\theta$.
- 20: Update the LRN parameters:
- 21: $\theta \leftarrow \theta - \alpha \nabla_{\theta} L_{oss}$.
- 22: **end for**

4. Experiments and Performance Evaluation

In this section, we establish a dataset and conduct experimental analyses, comparisons, and evaluations of recognition performance. Initially, we introduce the UWA communication system and the associated datasets used. Subsequently, using simulated data, we analyze and evaluate the relationship between recognition accuracy and SNR under conditions of insufficient labeled data through the SSL-LRN algorithm. Finally, using real data from pool and lake environments, we analyze and compare the recognition accuracy of the SSL-LRN algorithm with three other recognition algorithms under various SNR conditions.

4.1. UWA Communication System and Dataset

To establish a UWA communication recognition dataset in a real-world environment, we designed a broadband UWA communication system, the structure of which is shown in Figure 6.

In the transmission transducer, nine types of UWA communication signals are generated, including 2FSK, 4FSK, 8FSK, BPSK, QPSK, 8PSK, 16QAM, 64QAM, and OFDM. These signals are modulated by a computer and are stored as binary files. Subsequently, these binary files are transmitted to an NI USB-6259 data acquisition (DAQ) module and converted into analog signals. The analog signals are then amplified by a JYH500A linear power amplifier and transmitted through a WBT22-1107 acoustic transducer.

After transmission through the UWA channel, the signals are received by an RHS-20 hydrophone and amplified by a 2692-OS2 charge amplifier. The signals are then converted back into digital binary files by the USB-6259 and stored. To facilitate the identification of the start of the modulation signals, a synchronization signal is added.

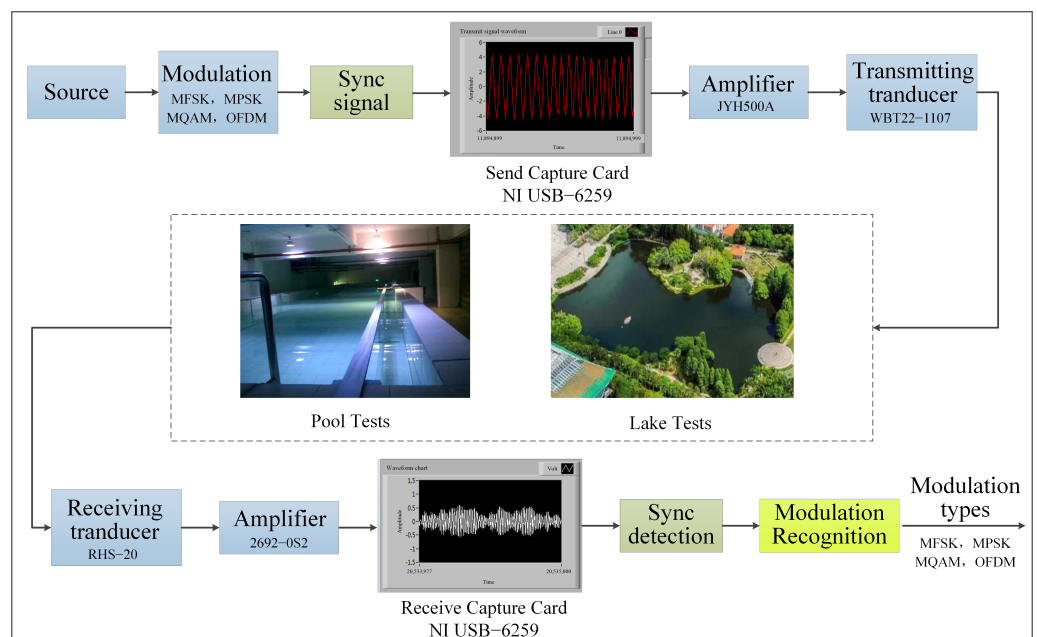


Figure 6. The broadband UWA communication system flowchart.

The dataset was collected during experiments conducted in both a swimming pool and a lake. In the pool experiments, the water depth was approximately 1.5 m, and communication distances ranged from 5 to 20 m. Some critical parameters of the lake experiments are shown in Table 1.

Table 1. Environmental parameters of lake tests.

Parameter Name	Parameter Value
Water depth	1–5 m
Distance between transmitting and receiving transducer	50–100 m
Distance between transmitting transducer and lake surface	1 m
Distance between receiving transducer and lake surface	3 m
Carrier frequency	10–15 kHz
Sample frequency	100 kHz
Number of samples	1100
Symbol rate	1000 Baud
Modulation types for transmission signal	2FSK, 4FSK, 8FSK, BPSK, QPSK, 8PSK, 16QAM, 64QAM, and OFDM

4.2. Performance Analysis with Insufficient Labeled Data

As shown in Figure 7, the relationship between recognition accuracy and SNR was examined using simulated data. Initially, the SSL-LRN algorithm was compared with its fully supervised version to assess the potential performance loss under conditions of insufficient labeled data. Subsequently, the SSL-LRN algorithm was compared with two other semi-supervised learning algorithms: SSRCNN [18] and SEMIAMC [19]. In Figure 7, the fully supervised version of the SSL-LRN algorithm is labeled as “Supervised (100%)”, indicating that 100% of the labeled data (1000 samples) were used during training. Similarly, “Supervised (10%)” indicates that only 100 labeled samples were used in training. In addition to “Supervised (100%)” and “Supervised (10%)”, all other algorithms utilized 100 labeled samples and 900 unlabeled samples during training.

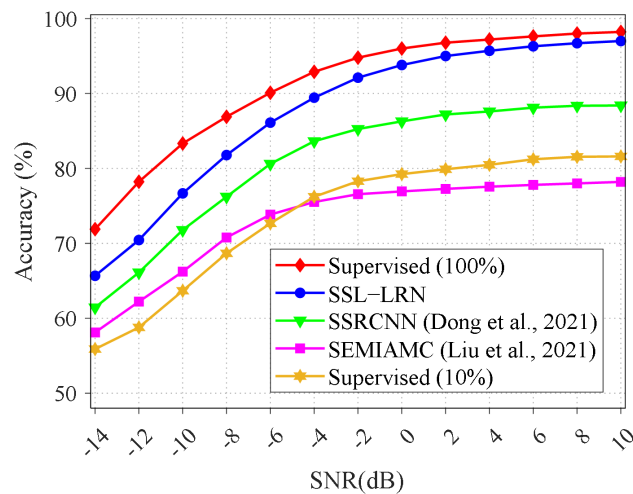


Figure 7. Recognition accuracy comparison for simulation channels: SSL-LRN, Supervised (100%), Supervised (10%), SSRCNN [18], and SEMIAMC [19].

4.2.1. Comparison with Other Semi-Supervised Algorithms

Among the three semi-supervised algorithms evaluated, the SEMIAMC algorithm exhibits the lowest recognition accuracy, closely aligning with “Supervised (10%)”.

Compared to the SEMIAMC algorithm, the SSRCNN algorithm shows a 9.3% higher recognition accuracy at an SNR of 0 dB. This performance improvement is attributed to its specially designed neural network architecture and loss function, which enhance its ability to learn from unlabeled data. However, the recognition accuracy of the SSRCNN algorithm is approximately 86.2% at an SNR of 0 dB, and it fails to reach 90% even at an SNR of 10 dB.

In contrast, the SSL-LRN algorithm performs optimally among these three semi-supervised algorithms. Compared to the SSRCNN algorithm, at an SNR of 0 dB, the SSL-LRN algorithm’s recognition accuracy is higher by 7.6%. This is due to the utilization of the mean squared error in the loss function of SSL-LRN, which is calculated as shown in (17) and improves the efficiency of using unlabeled data.

4.2.2. Analysis of Errors

The recognition accuracy of the SSL-LRN algorithm for each modulation type is displayed in Table 2. At 10 dB, most modulation types can be recognized with high accuracy, with precision rates nearing 100%. Even at −10 dB, the recognition accuracies for 2FSK, 4FSK, 8FSK, and OFDM remain above 96%. However, there were some errors in recognizing 16QAM and 64QAM at 10 dB, with accuracies of 81% and 88%, respectively. At −10 dB, primary recognition errors occurred in MPSK and MQAM.

Table 2. Detailed recognition accuracy of each modulation type.

Type	−14 dB	−12 dB	−10 dB	−8 dB	−6 dB	−4 dB	−2 dB	0 dB	2 dB	4 dB	6 dB	8 dB	10 dB
2PSK	30%	50%	73%	92%	98%	100%	100%	100%	100%	100%	100%	100%	100%
4PSK	52%	58%	62%	64%	66%	78%	87%	92%	98%	99%	100%	100%	100%
8PSK	22%	38%	43%	54%	54%	76%	86%	96%	97%	98%	100%	100%	100%
2FSK	92%	94%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
4FSK	75%	95%	98%	100%	98%	100%	100%	100%	100%	100%	100%	100%	100%
8FSK	91%	98%	99%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
16QAM	47%	54%	60%	68%	71%	72%	74%	76%	80%	80%	81%	82%	81%
64QAM	57%	77%	73%	78%	80%	84%	82%	85%	85%	86%	86%	87%	88%
OFDM	88%	91%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

To better analyze these errors, we utilized the confusion matrix shown in Figure 8. Within these matrices, two pairs of modulation types are primarily confused with each other: specifically, [16QAM, 64QAM] and [QPSK, 8PSK]. This confusion arises due to several factors. Primarily, the amplitude information of 16QAM and 64QAM is easily mixed under the influence of fading. Both 16QAM and 64QAM rely on variations in both amplitude and phase to encode information. In UWA channels, where signal strength can fluctuate significantly, the differences between these modulation types in terms of amplitude and phase become less distinct. This overlap makes it difficult for the recognition algorithm to accurately distinguish between the two modulation types, especially under conditions of severe fading.

Additionally, the small phase differences make it challenging to distinguish between QPSK and 8PSK in harsh UWA channels and low-SNR conditions. QPSK and 8PSK utilize phase shifts to encode data, but the phase angles between adjacent symbols in these modulation types are relatively close. In environments with significant noise and multipath effects, the received phase information can be distorted, leading to recognition errors. This distortion is particularly problematic at low SNRs, where the signal is already weak and any additional noise can further obscure the phase differences.

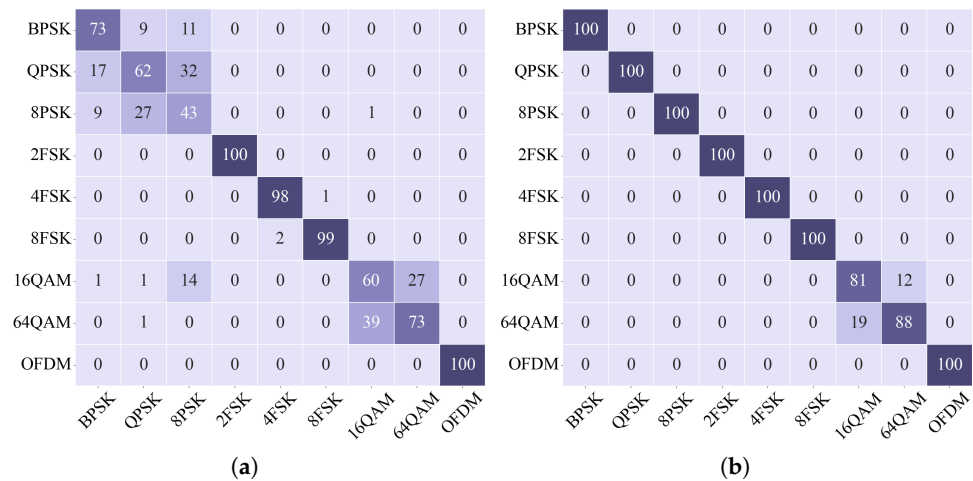


Figure 8. Confusion matrices at -10 dB and 10 dB. (a) -10 dB; (b) 10 dB.

However, no confusion occurred among MFSK, MPSK, or OFDM in high-SNR environments. These modulation types are inherently more robust against the specific types of distortions encountered in UWA channels. MFSK relies on frequency shifts, which are less susceptible to amplitude and phase noise, while OFDM’s use of multiple subcarriers allows for more reliable signal reconstruction even in the presence of multipath effects. Considering that 16QAM and 64QAM are typically used only in short-range, high-SNR underwater acoustic communication systems, the SSL-LRN algorithm still maintains high recognition accuracy in practical applications.

In summary, compared to other semi-supervised algorithms, the SSL-LRN algorithm performs exceptionally well. The recognition accuracy of the SSL-LRN algorithm is very close to that of fully supervised algorithms, even though it uses only 10% of the labeled data. The SSL-LRN algorithm can recognize most common types of UWA communications with high accuracy under low-SNR conditions.

4.3. Comparison with Other Recognition Algorithms

The performance of the SSL-LRN algorithm was also compared with three benchmark signal recognition algorithms. These include R&CNN [7], SSKNET [14], and IAFNET [9]. R&CNN integrates an RNN and a CNN to more effectively extract signal features. SSKNET, a kernel generation network, efficiently fuses signal features. IAFNET, based on denoising and task-driven mechanisms, improves recognition accuracy under low-SNR conditions. The size of the input data is closely related to the recognition latency and computational complexity. For example, R&CNN [7] requires an input size of 32×1280 , necessitating substantial FLOPs and resulting in increased recognition time delays. To facilitate a fair comparison of these algorithms’ performance, the signal input size was standardized to 64×64 .

4.3.1. Computation Complexity

The computational complexity of the SSL-LRN algorithm was analyzed in Table 3, which compares FLOPs, parameters, and memory across different algorithms. The SSL-LRN has the lowest computational complexity at 15.4 M FLOPs, mainly due to the quantization of the convolution kernel parameters. In addition, IAFNET’s FLOPs of 101.8 M are lower than those of R&CNN and SSKNET. Compared to IAFNET, SSL-LRN reduces FLOPs by 84.9%.

In terms of parameters and memory usage, IAFNET has the smallest values with 0.5 M and 1.9 M, respectively. SSKNET follows with 2.1 M parameters and 8.7 M memory usage. In contrast, the SSL-LRN has relatively higher parameters and memory usage, with 6.0 M and 17.3 M, respectively. This is mainly because the SSL-LRN uses channel expansion

modules and attention mechanisms. Although these increase the number of parameters and memory usage, they significantly improve recognition accuracy while maintaining low computational complexity.

FLOPs are a critical metric for determining whether an algorithm can be deployed on resource-limited devices, as they directly affect computational complexity and energy consumption. Although the SSL-LRN has relatively higher parameters and memory requirements, they are still within acceptable ranges, and the the SSL-LRN can efficiently run on resource-constrained devices.

Table 3. Computation complexity.

Network	FLOPs	Parameters	Memory
SSL-LRN	15.4 M	6.0 M	17.3 M
R&CNN [7]	240.0 M	4.6 M	17.8 M
SSKNET [14]	150.3 M	2.1 M	8.7 M
IAFNET [9]	101.8 M	0.5 M	1.9 M

4.3.2. Recognition Accuracy vs. SNR

The recognition accuracy of these algorithms was compared using simulated data as well as data from pool and lake environments. Pool and lake data were combined with simulated data to form the dataset used in this experiment. Only 10% of the data were used as labeled data. The relationship between recognition accuracy and the SNR is displayed in Figure 9.

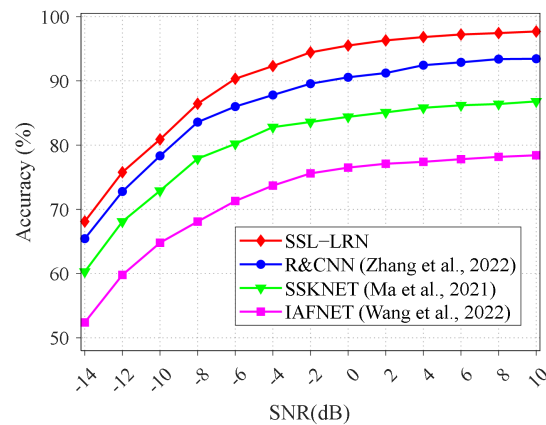


Figure 9. Recognition accuracy comparison for pool and lake channel: SSL-LRN, R&CNN [7], SSKNET [14], and IAFNET [9].

R&CNN utilizes RNN to extract temporal information from received signals. Compared to SSKNET and IAFNET, it demonstrates improved recognition performance. At 0 dB, the recognition accuracy of R&CNN reaches approximately 90.6%. Moreover, when the SNR exceeds 0 dB, this accuracy remains above 90%. Conversely, at the same SNR, SSKNET and IAFNET achieve lower recognition accuracies, falling below 90% and 80%, respectively.

However, R&CNN lacks attention mechanisms between convolutional layers and is not designed to efficiently utilize unlabeled data. In scenarios with insufficient labeled data, the recognition accuracy of R&CNN declines. The SSL-LRN algorithm exhibits the highest recognition accuracy. At 0 dB, using only 10% labeled data, the SSL-LRN achieves a recognition accuracy of approximately 95.5%. At 10 dB, the SSL-LRN’s recognition accuracy is about 97.7%. In summary, the SSL-LRN algorithm enhances recognition accuracy under low-SNR and insufficient-labeled-data conditions and can be applied to underwater embedded systems.

The performance of the SSL-LRN algorithm is assessed by analyzing its loss function values, which measure the algorithm’s learning capability. Figure 10 illustrates the variation of different types of loss function values across iterations on a simulated dataset.

Figure 10b,d clearly demonstrate that during the pre-training phase, despite a rapid decrease in labeled loss due to insufficient labeled data, the validation set loss fluctuates significantly, indicating poor performance and overfitting to the training data. Figure 10c shows that after the 10th epoch, with the influence of added unlabeled data, the loss function for unlabeled data L_u begins to significantly decrease, and the validation set loss stabilizes and eventually levels off around 0.05 after the 40th epoch. This indicates that the SSL-LRN algorithm effectively learns useful information from unlabeled data.

The total loss L_{oss} , as shown in Figure 10a, is calculated using Equation (18). Notably, the weight of the unlabeled data $r(t)$ increases over time. In our experiments, $r(t)$ is calculated using the following formula:

$$r(t) = \begin{cases} e^{-5(1-(t-10)/20)^2} & t \geq 10 \\ 0 & t < 10 \end{cases} \quad (19)$$

where t denotes the number of training epochs. Thus, after the 10th epoch, when unlabeled data are added, L_{oss} increases. However, as the loss from unlabeled data decreases, L_{oss} declines and ultimately converges to approximately 0.05, suggesting that the SSL-LRN algorithm learns effective features from unlabeled data, reducing reliance on limited labeled data and optimizing the overall learning process. This is supported by the results displayed for the SSL-LRN in Figure 7.

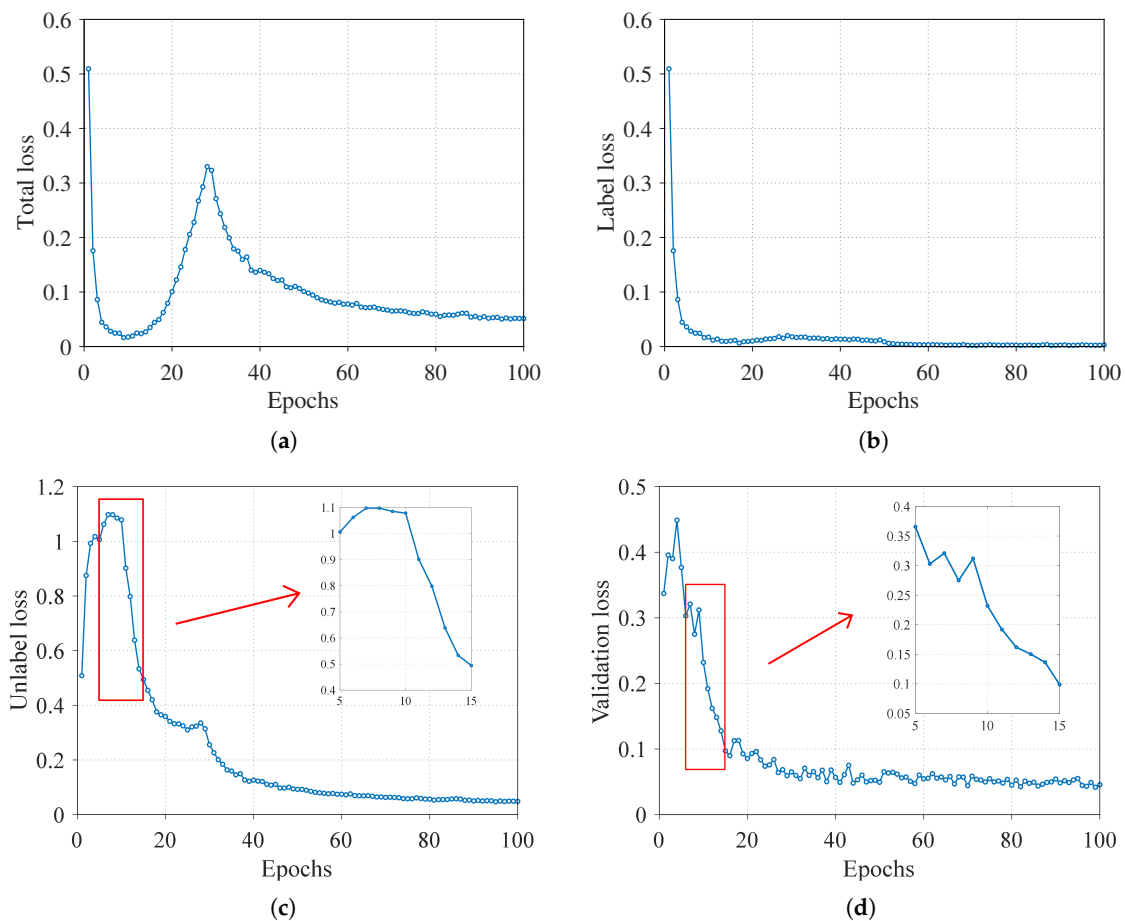


Figure 10. Loss functions of the SSL-LRN algorithm. (a) Total loss. (b) Labeled loss. (c) Unlabeled loss. (d) Validation loss.

5. Conclusions

Deep-learning-based algorithms have been extensively applied in communication modulation recognition and demonstrate superior recognition performance. However, these algorithms typically require large amounts of labeled data and have high computational complexity. Additionally, due to the multipath effect and Doppler effect in underwater channels, UWA signals experience severe frequency-selective fading. Therefore, achieving high recognition accuracy under low-SNR conditions is of great importance. This paper introduces the SSL-LRN algorithm for UWA modulation recognition; it is tailored to rapidly changing UWA channels and can be applied in practical scenarios such as underwater target detection and environmental monitoring.

Firstly, the SSL-LRN algorithm employs a lightweight network architecture that reduces FLOPs by 84.9% compared to baseline algorithms, making it more suitable for deployment in underwater devices. Secondly, the SSL-LRN algorithm can effectively utilize unlabeled data. Simulation results show that even with only 10% labeled data, the recognition accuracy of SSL-LRN is very close to that of fully supervised algorithms, with an average accuracy difference of only 3.2%. Lastly, the SSL-LRN algorithm achieves high recognition accuracy even at very low SNRs. Results from pool and lake tests demonstrate that even at an SNR of -4dB , the SSL-LRN achieves an average recognition accuracy of 92.3%, enabling the recognition of most commonly used UWA modulation types under low-SNR conditions.

In future work, we will further optimize the network architecture and develop adaptive mechanisms to enable the algorithm to automatically adjust parameters based on real-time channel conditions, thereby improving robustness in various environments.

Author Contributions: Conceptualization, C.D., W.S., and Z.X.; methodology, C.D., Z.X., and D.G.; software, C.D. and Z.X.; validation, C.D., W.S., and Z.X.; formal analysis, C.D., W.S., and E.C.; investigation, C.D., Z.X., and D.G.; resources, W.S. and E.C.; data curation, C.D. and Z.X.; writing—original draft preparation, C.D.; writing—review and editing, C.D., W.S., and Z.X.; visualization, C.D.; supervision, W.S. and E.C.; project administration, W.S.; funding acquisition, W.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (62071400).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are not publicly available due to privacy.

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

References

1. Luo, X.; Chen, L.; Zhou, H.; Cao, H. A Survey of Underwater Acoustic Target Recognition Methods Based on Machine Learning. *J. Mar. Sci. Eng.* **2023**, *11*, 384. [[CrossRef](#)]
2. Liu, F.; Zhang, Z.; Zhou, R. Automatic modulation recognition based on CNN and GRU. *Tsinghua Sci. Technol.* **2022**, *27*, 422–431. [[CrossRef](#)]
3. Xiong, W.; Bogdanov, P.; Zheleva, M. Robust and efficient modulation recognition based on local sequential IQ features. In Proceedings of the IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 1612–1620.
4. Li, Y.; Dou, Z.; Zhu, X.; Shi, C. Improving signal modulation recognition using principal component analysis and compressive sensing. In Proceedings of the IEEE INFOCOM Conference on Computer Communications Workshops, Honolulu, HI, USA, 15–19 April 2018; pp. 837–841.
5. Ge, Y.; Zhang, X.; Zhou, Q. Modulation Recognition of Underwater Acoustic Communication Signals Based on Joint Feature Extraction. In Proceedings of the 2019 IEEE International Conference on Signal, Information and Data Processing (ICSIDP), Zhuhai, China, 22–24 November 2019; pp. 1–4. [[CrossRef](#)]
6. Liu, B.; Jia, N.; Huang, J.; Guo, S.; Xiao, D.; Ma, L. Autoregressive model of an underwater acoustic channel in the frequency domain. *Appl. Acoust.* **2022**, *185*, 108397. [[CrossRef](#)]
7. Zhang, W.; Yang, X.; Leng, C.; Wang, J.; Mao, S. Modulation recognition of underwater acoustic signals using deep hybrid neural networks. *IEEE Trans. Wirel. Commun.* **2022**, *21*, 5977–5988. [[CrossRef](#)]

8. Yao, X.; Yang, H.; Sheng, M. Automatic Modulation Classification for Underwater Acoustic Communication Signals Based on Deep Complex Networks. *Entropy* **2023**, *25*, 318. [[CrossRef](#)] [[PubMed](#)]
9. Wang, H.; Wang, B.; Li, Y. IAFNet: Few-shot learning for modulation recognition in underwater impulsive noise. *IEEE Commun. Lett.* **2022**, *26*, 1047–1051. [[CrossRef](#)]
10. Gao, D.; Hua, W.; Su, W.; Xu, Z.; Chen, K. Supervised Contrastive Learning-Based Modulation Classification of Underwater Acoustic Communication. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1–10. [[CrossRef](#)]
11. Li, F.; Lin, X.; Shi, J.; Li, Z.; Chi, N. Modulation format recognition in a UVLC system based on reservoir computing with coordinate transformation and folding algorithm. *Opt. Express* **2023**, *31*, 17331–17344. [[CrossRef](#)]
12. Bai, H.; Huang, M.; Yang, J. An efficient Automatic Modulation Classification method based on the Convolution Adaptive Noise Reduction network. *ICT Express* **2023**, *9*, 834–840. [[CrossRef](#)]
13. Lyu, C.; Hu, X.; Niu, Z.; Yang, B.; Jin, J.; Ge, C. A light-weight neural network for marine acoustic signal recognition suitable for fiber-optic hydrophones. *Expert Syst. Appl.* **2024**, *235*, 121235. [[CrossRef](#)]
14. Ma, W.; Ma, H.; Zhu, H.; Li, Y.; Li, L.; Jiao, L.; Hou, B. Hyperspectral image classification based on spatial and spectral kernels generation network. *Inf. Sci.* **2021**, *578*, 435–456. [[CrossRef](#)]
15. Liu, J.; Li, T.; Xie, P.; Du, S.; Teng, F.; Yang, X. Urban big data fusion based on deep learning: An overview. *Inf. Fusion* **2020**, *53*, 123–133. [[CrossRef](#)]
16. Zhu, X.; Dong, H.; Salvo Rossi, P.; Landrø, M. Feature selection based on principal component regression for underwater source localization by deep learning. *Remote Sens.* **2021**, *13*, 1486. [[CrossRef](#)]
17. Lee, D.H.; Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In Proceedings of the Workshop Challenges Representation Learning, Atlanta, GA, USA, 17–19 June 2013; Volume 3, p. 896.
18. Dong, Y.; Jiang, X.; Cheng, L.; Shi, Q. SSRCNN: A semi-supervised learning framework for signal recognition. *IEEE Trans. Cogn. Commun. Netw.* **2021**, *7*, 780–789. [[CrossRef](#)]
19. Liu, D.; Wang, P.; Wang, T.; Abdelzaher, T. Self-contrastive learning based semi-supervised radio modulation classification. In Proceedings of the 2021 IEEE Military Communications Conference (MILCOM), San Diego, CA, USA, 29 November–2 December 2021; pp. 777–782.
20. Guo, Y.; Zhong, D.; Sun, H.; Jiang, Z.; Ye, L.; Deng, Z.; Liu, H. SemiAMR: Semi-Supervised Automatic Modulation Recognition With Corrected Pseudo-Label and Consistency Regularization. *IEEE Trans. Cogn. Commun. Netw.* **2024**, *10*, 107–121. [[CrossRef](#)]
21. Fu, X.; Peng, Y.; Liu, Y.; Lin, Y.; Gui, G.; Gacanin, H.; Adachi, F. Semi-supervised specific emitter identification method using metric-adversarial training. *IEEE Internet Things J.* **2023**, *10*, 10778–10789. [[CrossRef](#)]
22. Fang, T.; Wang, Q.; Zhang, L.; Liu, S. Modulation Mode Recognition Method of Non-Cooperative Underwater Acoustic Communication Signal Based on Spectral Peak Feature Extraction and Random Forest. *Remote Sens.* **2022**, *14*, 1603. [[CrossRef](#)]
23. Miao, Y.; Zakharov, Y.V.; Sun, H.; Li, J.; Wang, J. Underwater acoustic signal classification based on sparse time–frequency representation and deep learning. *IEEE J. Ocean. Eng.* **2021**, *46*, 952–962. [[CrossRef](#)]
24. Kaur, T.; Gandhi, T.K. Automated brain image classification based on VGG-16 and transfer learning. In Proceedings of the 2019 International Conference on Information Technology (ICIT), Bhubaneswar, India, 19–21 December 2019; pp. 94–98.
25. Liang, S.; Yin, S.; Liu, L.; Luk, W.; Wei, S. FP-BNN: Binarized neural network on FPGA. *Neurocomputing* **2018**, *275*, 1072–1086. [[CrossRef](#)]
26. Shen, M.; Han, K.; Xu, C.; Wang, Y. Searching for accurate binary neural architectures. In Proceedings of the IEEE/CVF International Conference Computer Vision Workshops, Seoul, Republic of Korea, 27–28 October 2019; pp. 2041–2044.
27. Cheng, P.; Liu, C.; Li, C.; Shen, D.; Henao, R.; Carin, L. Straight-through estimator as projected Wasserstein gradient flow. *arXiv* **2019**, arXiv:1910.02176.
28. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier nonlinearities improve neural network acoustic models. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 1–6.
29. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 448–456.
30. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 3–19.
31. Verma, V.; Kawaguchi, K.; Lamb, A.; Kannala, J.; Solin, A.; Bengio, Y.; Lopez-Paz, D. Interpolation consistency training for semi-supervised learning. *Neural Netw.* **2022**, *145*, 90–106. [[CrossRef](#)] [[PubMed](#)]
32. Tarvainen, A.; Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 1195–1204.
33. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D. Mixup: Beyond empirical risk minimization. *arXiv* **2017**, arXiv:1710.09412.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.