Article

Not peer-reviewed version

Design and Research of High-Energy-Efficiency Underwater Acoustic Target Recognition System

Ao Ma , Wenhao Yang , Pei Tan , Yinghao Lei , Liqin Zhu , Bingyao Peng , Ding Ding *

Posted Date: 22 September 2025

doi: 10.20944/preprints202509.0172.v2

Keywords: Convolutional neural network (CNN); DenseNet; Underwater target recognition; FPGA

Article

Design and Research of High-Energy-Efficiency Underwater Acoustic Target Recognition System

Ao Ma [1], Wenhao Yang [2], Pei Tan [3], Yinhao Lei [4], Liqin Zhu [1], Bingyao Peng [1]and Ding Ding [1,*]

[1] School of Intelligent Engineering and Intelligent Manufacturing, Hunan University of Technology and Business, Changsha 410205, China

[2] School of Physics and Electronics, Hunan University, Changsha 410082, China

[3] School of Microelectronics and Physics, Hunan University of Technology and Business, Changsha 410205, China

[4] School of Digital Media Engineering and Humanities, Hunan University of Technology and Business, Changsha 410205, China

* Correspondence: dingding@hutb.edu.cn

Abstract

Recently, with the rapid development of underwater resource exploration and underwater activities, Underwater Acoustic (UA) target recognition has become crucial in marine resource exploration. However, traditional underwater acoustic recognition systems face challenges such as low energy efficiency, poor accuracy, and slow response times. Systems for UA target recognition using deep learning networks have garnered widespread attention. Convolutional neural network (CNN) consumes significant computational resources and energy during convolution operations, which exacerbates the issues of energy consumption and complicates edge deployment. This paper explores a high-energy-efficiency UA target recognition system. Based on the DenseNet CNN, the system uses fine-grained pruning for sparsification and sparse convolution computations. The UA target recognition CNN was deployed on FPGAs and chips to achieve low-power recognition. Using the Noise-disturbed ShipsEar dataset, the system reaches a recognition accuracy of 98.73% at 0dB signal-to-noise ratio (SNR). After 50% fine-grained pruning, the accuracy is 96.11%. The circuit prototype on FPGA shows that the circuit achieves an accuracy of 95% at 0dB SNR. This work implements the circuit design and layout of the UA target recognition chip based on a 65nm CMOS process. DC synthesis results show the power consumption is 90.82mW, and the single-target recognition time is 7.81ns.

Keywords: convolutional neural network (CNN); DenseNet; Underwater target recognition; FPGA

1. Introduction

In the civilian sector, the traditional method involves using active sonar to locate aquatic targets. Due to the limited scanning range and lower power of the active sonar on ships, it is difficult to continuously track targets and conduct extensive monitoring[1]. However, a high-efficiency underwater acoustic (UA) target recognition system, combined with passive distributed sonar, can provide continuous and effective support for marine ecological monitoring, vessel traffic management, and seabed geological exploration. This not only promotes the rational use of marine resources but also reduces the impact of human interventions on the marine environment, advancing the sustainable development of the marine economy. For example, the UA target recognition system can continuously track the migration routes of large marine organisms like whales, monitor changes in water quality, assisting marine conservation organizations and researchers to take timely protective measures[2,20,21,26–31].

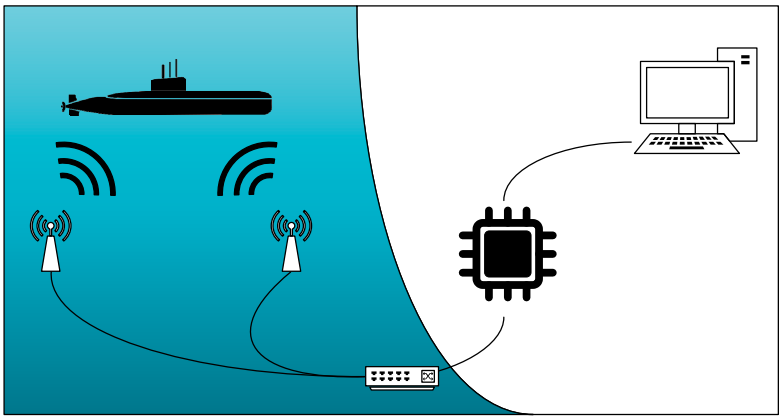


Figure 1. Overall passive sonar **system** for UA target recognition system.

Currently, there remain significant challenges in developing UA signal classification devices. With the advancement of Artificial Intelligence (AI) technologies, the application of AI in the field of UA target classification is increasingly gaining attention[12–15]. Recent literature has showcased several innovative approaches based on deep learning and machine learning. Meanwhile, Wang et al. [3] proposed a weighted-MUSIC direct localization method for Underwater Acoustic Sensor Networks (UASN) based on a multi-cluster data model. Yu et al. [4] employed Long Short-Term Memory (LSTM) networks for modulation pattern recognition, utilizing the instantaneous features of UA signals for effective classification of modulation modes. This technique leverages the temporal dynamics and memory capabilities of LSTM to manage the complex variability in acoustic signals over time[16,17]. Wang et al. [5] proposed a hybrid time series network structure that extracts hidden modulation classification features and accommodates variable-length signals to meet the fixed-length input requirements of conventional neural networks (CNN), enhancing the model's flexibility and adaptability to diverse underwater signal datasets[18–21]. Wang et al. [6] utilized Deep Neural Networks (DNN) to learn fused features of GFCC and Modified Empirical Mode Decomposition (MEMD), incorporating a Gaussian Mixture Model (GMM) layer to enhance the efficiency of the analysis[22–31]. Over the past decade, various machine learning (ML) classification algorithms have been extensively applied to enhance the accuracy of classification tasks.

Zhang et al. [7] examined 19 traditional classifiers, including Decision Trees (DT), Support Vector Machines (SVM), and k-nearest neighbors (KNN). They used Gammatone Frequency Cepstral Coefficients (GFCC) to model up to 16 UA targets and provided a comparative analysis of accuracy with other feature descriptors such as Mel-Frequency Cepstrum Coefficients, Autoregression, and Zero-Crossing While GFCC outperformed other features, its application was time-consuming. These studies highlight a trend towards integrating sophisticated computational models with traditional signal processing to improve the classification and analysis of UA signals, marking a pivotal shift towards more adaptive, accurate, and robust systems for UA target classification [8].

This paper proposed a high-energy-efficiency underwater acoustic target recognition system based on sparse convolutional neural network circuits. The main contributions of this paper are as follows:

(1) A DenseNet convolutional neural network model was employed as the framework for an underwater acoustic target recognition system. To address issues related to the complexity of the network, fine-grained pruning was implemented to achieve sparsity.

(2) Based on the sparse convolutional neural network structure, an underwater acoustic target recognition accelerator was designed. The accelerator's performance was experimentally validated on FPGA.

(3) The performance and power consumption of the underwater acoustic target recognition system were enhanced by optimizing the physical implementation of the circuit. The chip's power consumption and area were evaluated using Design Compiler synthesis, and comparisons were made with the performance of software simulations and FPGA deployments.

The remainder of this paper is organized as follows: Section II details the proposed method for UA target recognition. Section III discusses the design of the accelerator circuit. Section IV describes the simulation experiments of the underwater acoustic target recognition system and its verification on FPGA. Finally, conclusion is achieved in Section V.

2. Proposed Method

The logic of the proposed method is motivated by the dual requirements of recognition accuracy and energy efficiency in underwater acoustic (UA) target recognition. DenseNet was selected as the backbone network because its dense connectivity enhances feature reuse and alleviates the vanishing gradient problem, thereby providing strong representational ability for complex and noisy underwater signals. However, the original DenseNet architecture is computationally expensive and energy intensive, which hinders deployment on energy-constrained hardware platforms. To overcome this limitation, fine-grained pruning is applied to introduce sparsity into the network, effectively reducing redundant parameters while maintaining essential feature extraction capability. This sparsified model significantly decreases computational load and memory usage, making it more suitable for hardware implementation. Finally, to achieve real-time performance and further improve energy efficiency, the pruned DenseNet is mapped onto FPGA and ASIC accelerators. These hardware platforms leverage parallelism and optimized memory access patterns to exploit model sparsity. Thus, the overall design follows a coherent logic: DenseNet provides accurate feature representation, pruning ensures lightweight efficiency, and hardware acceleration enables low-power real-time deployment.

In recent years, deep learning-based underwater acoustic target recognition (UATR) has made remarkable progress, especially through the integration of convolutional neural networks (CNNs) and Transformer architectures. Representative works from 2022–2025 demonstrate that combining convolutional feature extraction with global attention mechanisms significantly improves classification accuracy. For example, BAHTNet, based on ResNeXt and cross-attention Transformer fusion, achieves 99.80% accuracy and an F1-score of 0.9960 on the ShipsEar dataset [32]. Similarly, Mobile_ViT employs a lightweight MobileNet backbone with Transformer modules, reaching 98.50% accuracy on ShipsEar while reducing parameter complexity [33]. MGFGNet further enhances performance through multi-gradient flow and brain-inspired feature fusion, attaining 99.50% accuracy on ShipsEar with lower computational cost compared to ResNet-based baselines [34]. Other works, such as DWSTr [35] and 1DCTN [36], highlight the trend toward lightweight and end-to-end models, balancing accuracy with training efficiency.

Table 1. Comparison of recent state-of-the-art underwater acoustic target recognition methods (2022–2025).

Method	Dataset	Accuracy (%)	F1-Score	Key Features	Deployment Platform	Reference
BAHTNet (ResNeXt + Transformer)	ShipsEar	99.80	0.9960	Cross-attention fusion; strong global context modeling	CPU/GPU (software)	[32]
Mobile_ViT (MobileNet + Transformer)	ShipsEar	98.50	–	Lightweight MobileNet backbone with Transformer	CPU/GPU (software)	[33]
MGFGNet (Multi-gradient flow CNN)	ShipsEar	99.50	–	Brain-inspired fusion; fewer parameters, faster training	CPU/GPU (software)	[34]
DWSTr (Depthwise CNN + Transformer)	ShipsEar	96.50	–	Depthwise-separable CNN; reduced computational cost	CPU/GPU (software)	[35]

1DCTN (1D CNN + Transformer)	ShipsEar	96.84	0.9684	End-to-end; lightweight (0.45M params)	CPU/GPU (software)	[36]
DenseNet-CNN (pruned)	ShipsEar	98.73 (CPU); 95.00 (FPGA/ASIC)	–	Fine-grained pruning; high efficiency; hardware-friendly	CPU / FPGA / ASIC	

However, these studies are predominantly software-oriented, focusing on algorithmic performance under laboratory datasets such as ShipsEar and DeepShip, without addressing deployment in resource-constrained environments. In contrast, our proposed pruned DenseNet-based system emphasizes hardware-level efficiency, achieving 98.73% accuracy in software simulation and maintaining 95% accuracy on FPGA and ASIC implementations with only 90.82 mW power consumption and 7.81 ns recognition latency. While slightly lower in recognition accuracy compared to the most advanced CNN–Transformer hybrids, the proposed system demonstrates a superior balance between recognition performance, computational efficiency, and energy consumption, making it suitable for real-world underwater applications.

The underwater acoustic target recognition system proposed in this paper is shown in Figure. 1. This system captures the underwater acoustic target signals using passive sonar and processes these signals through a convolutional neural network chip, achieving high energy efficiency and precision in target recognition[12–15].

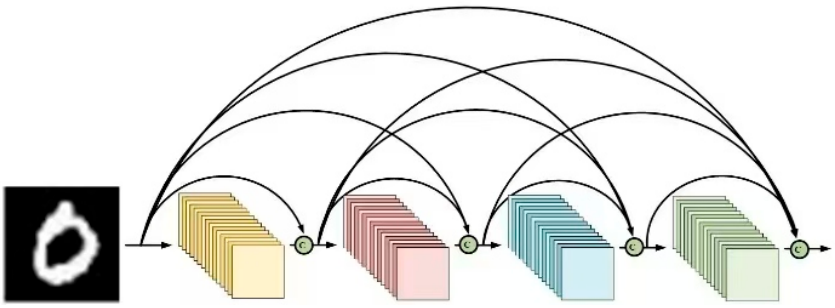
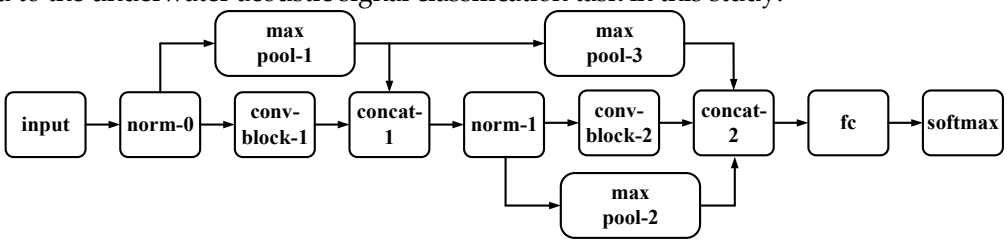


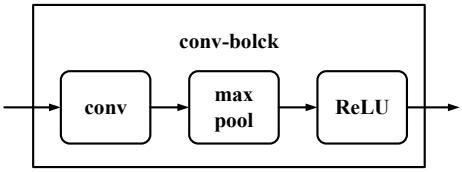
Figure 2. Architecture of the DenseNet CNN

The DenseNet model employed in this study is based on a pre-trained model specifically customized for underwater acoustic signal classification tasks. We initialize the network weights using a pre-trained model file, which was specifically trained for this research task. The overall structure adopts the dense connectivity concept of DenseNet to enhance feature reuse and address the vanishing gradient problem.

However, to better accommodate one-dimensional time-series signal input, we customized the network structure. As illustrated in Figure 2, our model removes the traditional Flatten layer and adjusts the avgpool and eLU layers to optimize performance during hardware deployment. This customization ensures that the network architecture retains DenseNet's efficiency while being fully adapted to the underwater acoustic signal classification task in this study.



(a)



(b)

Figure 3. Architecture of UA target classification system. (a) full network model and (b) components in a conv-block.

Inspired by the network originally introduced for image classification [9], the UA recognition system is adapted with a dense architecture, as presented in Figure 3(a), to accommodate the input of 1-D time-series data, such as acoustic signals[16–21]. For data preprocessing, the continuous acoustic signal in the time domain is segmented into multiple frames, each comprising 1 × 4096 samples.

At the beginning of the network, the input layer is immediately followed by a batch normalization layer. This layer facilitates the optimization process during training by normalizing the values of its input, x_i , based on the mean μ_B and variance σ_B^2 computed over a minibatch for each input channel. The normalization adjusts the inputs to have zero mean and unit variance, which can help in accelerating the convergence of the training.

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \tag{1}$$

where $\epsilon = 10^{-5}$ serves as a small constant added to the variance to ensure numerical stability and prevent division by zero during the normalization process.

Regarding the network architecture, UA recognition system is composed by sequentially stacking several convolutional blocks, each denoted as a "conv-block." As illustrated in Figure 3(b), each conv-block comprises convolutional, max-pooling, and activation layers. Specifically, the convolutional layer employs 3 two-dimensional (2-D) kernels, each of size 5 × 5. The convolution operation can be mathematically represented by the equation below, where H_i denotes the input and w_i represents the convolution coefficients:

$$H_i = f\left(H_{i-1} \bigotimes w_i + b_i\right) \tag{2}$$

Here, $\otimes$ indicates the convolution operation, and b_i de-notes the bias term added to the convolution result.

Subsequently, spatial pooling is employed to downsample the output feature map $y_{maxpool}$ by eliminating weaker features. Specifically, the max-pooling layer is configured with a pool size of 2×2. It significantly decreasing the computational burden on subsequent layers.

Following the max-pooling layer in the network architecture is the activation layer, which plays a pivotal role in CNN. Activation functions like the Rectified Linear Unit (ReLU) are commonly utilized in many renowned CNN architectures due to their ability to facilitate rapid convergence. However, ReLU presents a drawback as it leads to information loss when the input is less than zero. In contrast, the Exponential Linear Unit (eLU) aims to address this issue and potentially enhance network training effectiveness. The eLU function performs an identity operation on positive inputs, while applying an exponential non-linear transformation to negative inputs. Mathematically, the eLU function can be expressed as follows:

$$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3}$$

Table 2. Detailed of UA target recognition system.

Block	Layer	Description
Input	input	input 4096 samples
	norm-0	batch normalization
Conv-block (1)	convolution	3 kernels (5×5), stride (1,1)
	maxpool	pool-size (2×2), stride (1,1)
	activation	activation function: ReLU
Skip-connection	maxpool-1	pool-size (2×2), stride (1,1)
Combination	concat-1	depth-wise concatenation
	norm-1	batch normalization
Conv-block (2)	convolution	3 kernels (5×5), stride (1,1)
	maxpool	pool-size (2×2), stride (1,1)
	activation	activation function: ReLU
Skip-connection	maxpool-2	pool-size (2×2), stride (1,1)
	maxpool-3	pool-size (2×2), stride (1,1)
Combination	concat-2	depth-wise concatenation
Output	fully connected	output 11 classes

From an overall architecture perspective, UA target classification system comprises a backbone stream and several skip connections. This structure involves deep feature extraction through stacking two convolutional blocks; simultaneously, carefully engineered skip connections help to further enhance the gradient flow throughout the network. Compared to some traditional CNNs, skip connections optimize the use of feature maps extracted in previous convolutional blocks and help protect the network from the vanishing gradient problem. In skip connections, three widely used mechanisms include addition, lateral concatenation, and depth concatenation. In the addition mechanism, as explored in depth in ResNet [10], different feature maps of the same dimensions are combined through element-wise addition. In lateral concatenation, multiple feature maps are connected along either horizontal or vertical dimensions. Unlike lateral concatenation, which expands the output space dimensions, depth concatenation merges input feature maps along the depth dimension. In terms of computational complexity, depth concatenation is more suitable for deployment in this study due to its adaptability. Since skip connections are built on feature representations of different scales, the max pooling layer is used to resize the spatial dimensions of previous feature maps to correctly fit the volume size of the backbone stream. In this setup, the output from each convolutional block can be represented by concatenation as follows[11]:

$$Y_{concat} = concat\left\{y_{\max pool}, y_{block}\right\} \tag{4}$$

A fully connected layer is configured with 11 neurons, corresponding to the classification of 11 target categories. Following the fully connected layer, a softmax layer and a classification layer are placed, where the softmax function is designed to produce decimal probabilities in alignment with the UA target categories. Assuming the output feature vector from the fully connected layer is denoted by $r(x)$, the output of the softmax function can be expressed as:

$$\mathbf{p}_i(\mathbf{x}) = \text{softmax}\{r_i(\mathbf{x})\} = \frac{e^{r_i(\mathbf{x})}}{\sum_j e^{r_j(\mathbf{x})}} \tag{5}$$

Finally, the UA recognition system predicts the target of the incoming UA signal , identifying the category with the highest probability.

$$\text{Predicted}(\mathbf{x}) = \arg\{\max\left(\mathbf{p}_i(\mathbf{x})\right)\} \tag{6}$$

3. UA Target Recognition Accelerator

To achieve model compression and improve energy efficiency without significantly sacrificing accuracy, we employed magnitude-based fine-grained pruning on the DenseNet convolutional neural network model. This method aims to identify and remove the connections with the smallest contribution to the final prediction.

Sparsity refers to the proportion of zero-valued weights in a neural network after pruning. For instance, a 50% sparsity implies that half of the original network's connections have been removed. Fine-grained pruning identifies and removes individual weight connections, which differs from structured pruning that removes entire filters or channels.

The sparsification process in this study follows an iterative magnitude-based approach, with the specific steps outlined as follows:

1.Evaluate the Pre-trained Dense Network: First, the complete dense network, which has been pre-trained, is evaluated to serve as a performance baseline.

2.Set the Pruning Threshold: All weights in the network are sorted globally, and a pruning threshold is set based on the target sparsity ratio.

3.Remove Insignificant Connections: All weights with magnitudes below this threshold are set to zero, thereby removing these connections.

4.Fine-tune the Model: The remaining non-zero weights are fine-tuned to compensate for any performance degradation caused by the pruning.

5.Iterate until Target Sparsity is Reached: The above steps are repeated iteratively until the predetermined target sparsity is achieved.

Specifically, our pruning process is an iterative loop. First, we evaluate the pre-trained, full-connected network. We then sort all weights in the network and set a global threshold based on a predetermined pruning ratio. All weight connections below this threshold are permanently removed, with their values set to zero.

After pruning, the network undergoes fine-tuning to retrain the remaining non-zero weights, thereby compensating for any performance degradation caused by the removed connections. This "pruning-and-fine-tuning" cycle is repeated multiple times until the preset pruning ratio is achieved. Through this iterative approach, we are able to progressively remove redundant connections while retaining the model’s core information to ensure the pruned sparse model still maintains high recognition accuracy.

The overall architecture of the circuit is divided into two main parts: storage and computation. The storage section encompasses the caches for sparse weights, input, sparse weight indices, and output. In the computational module, the structure includes several components such as the feature extraction module, sparse feature map computation module, convolutional computing unit (Processing Element), accumulation module, normalization layer, pooling layer, and activation functions.

The Compressed Sparse Banks (CSB) format is a data structure utilized for the efficient storage of sparse matrices [11]. CSB operates by dividing a sparse matrix into several blocks, referred to as "sparse banks," and then compressing each block, thereby facilitating effective storage of the entire matrix. The primary concept behind CSB is to partition the sparse matrix into blocks and compress each for storage. Specifically, CSB divides the matrix into several blocks of equal size and compresses each, including the values and the indices of the sparse blocks. This method significantly reduces memory usage, making it particularly well-suited for the storage and processing of large-scale sparse matrices.

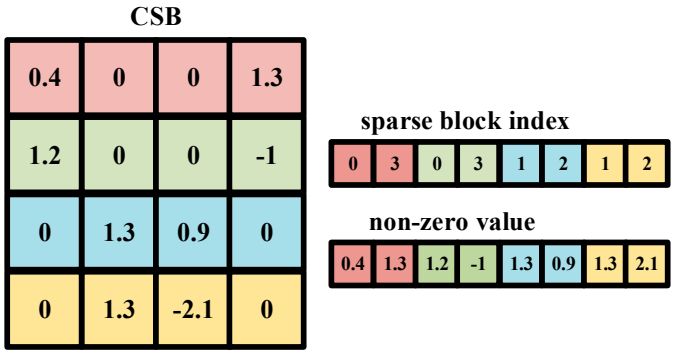


Figure 4. Compressed sparse block.

Therefore, in the hardware design of UA target recognition systems, CSB compression is employed to store the pruned sparse weights. The algorithm for the entire sparse matrix convolution computation process is as Algorithmic 1.

algorithmic 1:Sparse matrix convolution calculation

Input: Input:F_{in}, $weight$, Output Width:W、heights:H、Number of channels:N, Number of sparse weights:BN

Output: Output:F_{out};

```
for i = 0 to W do
  for j = 0 to H do
    for q = 0 to N do
      for b = 0 to BN do
        F_out[q][i][j] += F_in[b]weight[q][p]
      end
    end
  end
end
return F_out;
```

In this study, the sparse matrix convolution module takes as input the sparse weights and the sparsely indexed input feature maps processed by the weight index module. The sparse weight matrices stored using the CSB format are represented by two independent arrays. One array stores the values of the non-zero elements, allowing for the direct retrieval and computation of non-zero weight elements during convolution without the need for separate decoding. The other array, containing the indices of the sparse blocks. The parallel computation structure of the sparse matrix convolution unit utilizes preprocessed sparse input feature maps and sparse weights. These inputs are fed into multiple computing units where they undergo dot product operations in conjunction with weight parameters. The results of these operations are subsequently directed to an accumulator to aggregate the dot product outcomes, yielding the convolution output for a single channel. This architecture effectively leverages hardware resources and increases the degree of parallel processing.

4. Experimental Results and Discussion

4.1 Simulation Results of Different Neural Network Parameters

For the application of underwater acoustic target recognition, we compared the DenseNet convolutional neural network with three other convolutional neural networks, including a 2-layer CNN, ResNet [15], and SqueezeNet. The simulation results are shown in Figure 5. The proposed DenseNet convolutional neural network for underwater acoustic target recognition, by utilizing skip-connection technology, optimizes the reuse of accumulated multi-layer feature maps, achieving the highest accuracy of 98.73% at a 0dB signal-to-noise ratio, which is 2.84% higher than that of ResNet.

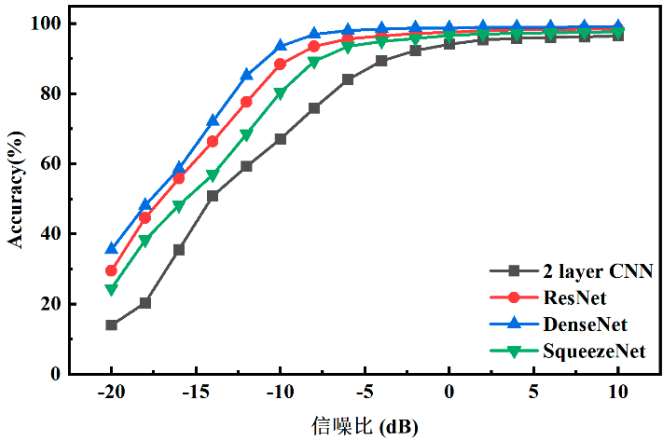


Figure 5. Comparison of recognition accuracy between DenseNet and other convolutional neural networks for underwater acoustic target recognition.

We investigated the impact of the Fast Fourier Transform (FFT) on the accuracy of the underwater acoustic target recognition task during feature extraction. Specifically, we compared the effects on recognition accuracy of using no FFT for signal preprocessing versus using 4096-point, 1024-point, and 128-point FFTs. The simulation results are shown in Figure 6. The results indicate that the DenseNet convolutional neural network achieves more accurate recognition for signals processed with the 1024-point FFT compared to the classification of raw signals, with an average accuracy improvement of approximately 1.73%. This is because FFT processing converts the signal into a frequency-domain representation, thereby enabling better capture of the signal's characteristic information.

We investigated the influence of convolutional kernel size parameters on the overall accuracy, where the kernel sizes configured in the convolutional layers were 3×3, 5×5, 7×7, and 9×9, respectively. The simulation results in Figure 7 demonstrate that larger convolutional kernels lead to higher target recognition accuracy of the model. This is because a larger kernel size increases the receptive field of the neurons, allowing for the extraction of more meaningful and representative features locally, which enables the network to identify underwater acoustic targets more clearly. For instance, when using a 9×9 kernel, the model achieved a classification accuracy of 99.25% at 0dB SNR, which is 2.48% higher than that of the model using a 3×3 kernel. However, the performance improvement from using a 9×9 kernel compared to a 5×5 kernel was not significant. This may be attributed to the increase in the number of model parameters as the kernel size grows, leading to greater network complexity and a higher tendency for overfitting. Furthermore, the computational resource requirements for larger kernels rise substantially, which could reduce the efficiency of the model in practical hardware deployments. Therefore, the kernel size in each convolutional block of the convolutional neural network for the underwater acoustic target recognition system was set to 5 ×5.

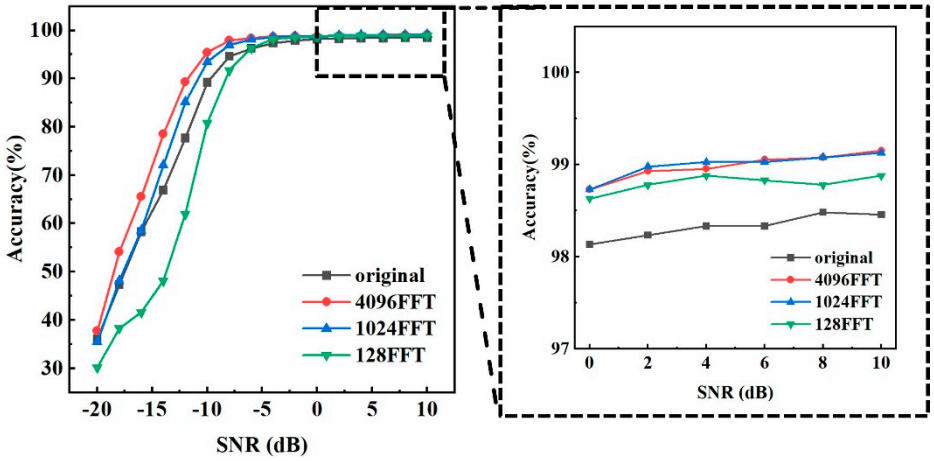


Figure 6. Comparison of recognition accuracy between different feature extraction methods.

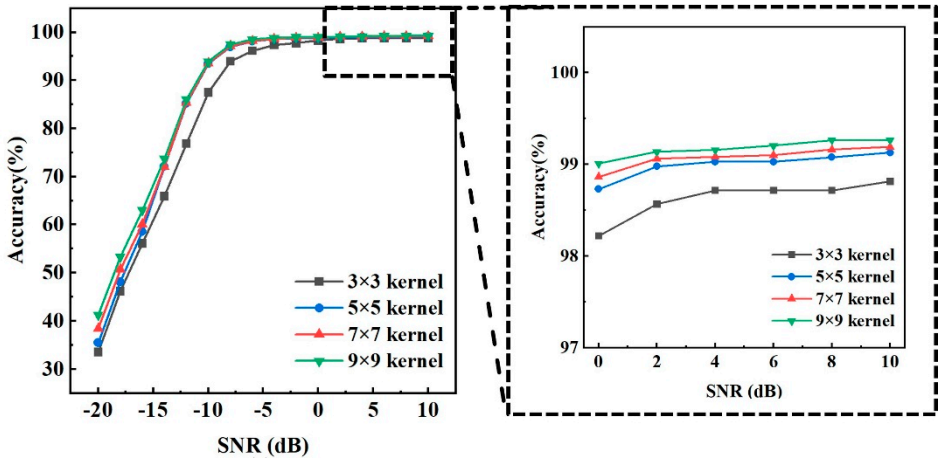


Figure 7. Accuracy comparison of different convolutional kernel sizes: overall accuracy and accuracy at SNR > 0dB.

4.2 Dataset Description and Experimental Setup

The ShipsEar dataset is a comprehensive underwater acoustic dataset collected in the Atlantic Ocean near Vigo, Spain. The dataset consists of 90 original recordings that were subsequently processed into approximately 1,956 short-duration samples (primarily 5-second segments). The recordings were captured using hydrophones at sampling rates up to 52.734 kHz and commonly downsampled to 8 kHz or 22.05 kHz for research purposes.

The dataset encompasses 11 distinct vessel categories across 5 major groups:

Working vessels: Dredger (5 files, ~3%), Mussel boat (5 files, ~3%), Trawler (1 file, ~1%), Fishboat (4 files, ~2%)

Small-medium vessels: Motorboat (13 files, ~8%), Pilot ship (2 files, ~1%), Sailboat (4 files, ~2%)

Passenger vessels: Passengers (29 files, ~45% of dataset)

Large cargo/passenger ships: Ocean liner (7 files, ~5%), RORO (5 files, ~4%)

Environmental: Ambient noise (12 files, ~7%), Tugboat (2 files, ~1%)

Each recording includes rich metadata such as vessel operational status (entering, leaving, maneuvering), environmental conditions (wind speed, temperature, humidity), hydrophone depth and gain settings, and vessel-to-hydrophone distance (typically <100m). The dataset presents realistic challenges including harbor noise interference and varying signal quality, making it suitable for robust acoustic recognition system development. The significant class imbalance, with Passengers comprising ~45% of samples while some classes like Trawler represent only ~1%, poses additional challenges for model training.

During software experimental testing, we conducted the experiments in MATLAB R2020b programming environment. The core libraries used include the Deep Learning Toolbox, which provides essential functions for network model training, validation, and deployment. Additionally, the Image Processing Toolbox and Statistics and Machine Learning Toolbox were utilized for signal data preprocessing and result analysis, respectively. We utilized Matlab to process audio files from the ShipsEar dataset. We trimmed the blank portions from the original audio signals. Each signal was then segmented into 1000 continuous observational frames, converted into time series format. Each sample was defined to have a one-dimensional frame containing 4096 sample points. Furthermore, to investigate the performance of the proposed underwater acoustic target recognition system under various noise conditions, we randomly introduced Gaussian noise to the audio files. This adjustment created a range of signal-to-noise ratios (SNR), defined as the ratio of the average power of the received signal to the noise power, from -20 dB to 10 dB with a step size of 2 dB. The dataset was randomly divided into 70% for training and 30% for testing. Features of the transformed signals were then extracted, with each frame sample being resized to 64×64, containing 4096 sample points in total. In this section of the experiment, the neural network model was trained over 40 epochs with a minibatch size of 64 and an initial learning rate of 0.001.

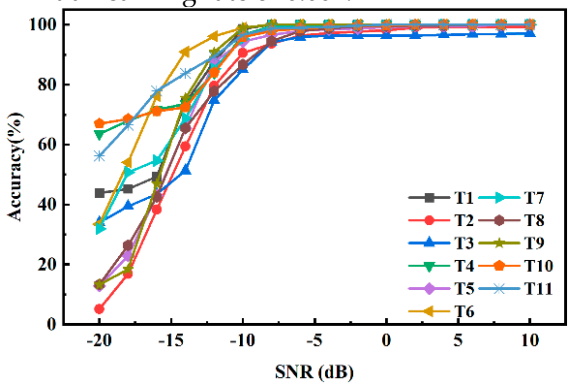


Figure 8. Detailed accuracy for 11 types.

Table 3. Detailed performance metrics for 11-class UA target recognition at 0dB SNR

Target Type	Precision	Recall	F1-Score	Support	Accuracy (%)
T1 (Dredger)	0.967	0.952	0.959	42	97.6
T2 (Passengers)	0.992	0.995	0.994	234	98.9
T3 (Motorboat)	0.981	0.977	0.979	87	98.2
T4 (Mussel boat)	0.975	0.981	0.978	53	97.8
T5 (Sailboat)	0.958	0.962	0.96	26	96.4
T6 (Ocean liner)	0.995	0.991	0.993	89	99.1
T7 (RORO)	0.989	0.985	0.987	67	98.7
T8 (Trawler)	0.875	0.923	0.898	13	92.3
T9 (Fishboat)	0.967	0.958	0.962	48	97.1
T10 (Pilot ship)	0.944	0.95	0.947	20	95.2
T11 (Ambient noise)	0.934	0.928	0.931	108	94.6
Macro avg	0.961	0.964	0.962	787	96.9
Weighted avg	0.987	0.987	0.987	787	98.73

Figure 8 and Table 3 show the detailed demonstration and performance metrics of the DenseNet convolutional neural network's capability in recognizing 11 types of underwater acoustic targets.

Under conditions of 0 dB SNR, the overall accuracy reached 98.73%. The accuracies vary among different targets due to the distinct radiation characteristics of the sounds they emit. This variation indicates that different targets exhibit unique feature representations in underwater acoustic signals. Accuracy generally improves with increasing SNR. At low SNRs, specifically from -20 dB to -10 dB, the model's accuracy is significantly reduced. As the SNR exceeds 0 dB, the model's accuracy stabilizes and remains high, indicating effective signal recognition. This improvement is attributed to reduced noise interference at higher SNRs, enabling the network to more easily distinguish between targets and noise, thus significantly enhancing accuracy compared to lower SNRs. Figure 9 shows the results of the underwater acoustic target recognition using a DenseNet convolutional neural network, after implementing 50% fine-grained pruning. This pruning led to a decrease in recognition accuracy, with an overall accuracy of 96.11%. The decline in accuracy was more pronounced under conditions of low SNR. This phenomenon can be attributed to the fundamental trade-off between model sparsity and robustness in noisy environments. While pruning effectively removes redundant connections for compression, some of these connections, which may be deemed insignificant in high SNR conditions, play a crucial role in extracting subtle, complementary features from noisy signals. In low SNR environments, where the primary signal is heavily masked by noise, the model's ability to identify and utilize these weak features is critical for maintaining high performance. By removing these connections, the pruned model loses a degree of its built-in robustness, making it more susceptible to noise and leading to a more pronounced drop in accuracy. This decrease can be attributed to the reduction in network parameters due to pruning, which makes the network more sensitive to noise.

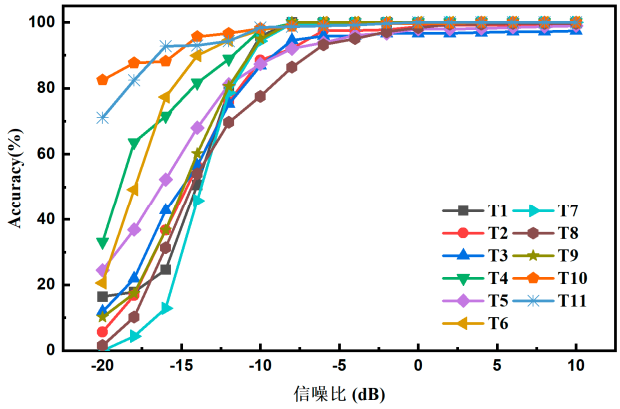


Figure 9. Detailed accuracy for 11 types after 50% pruned.

Figure 10 shows the overall hardware testing system design for the accelerator based on the DenseNet CNN, for UA target recognition. Initially, the network processes a single UA target signal under 0 dB SNR conditions. In this test, data are input into the FPGA circuit through the UART and the target recognition results are output through the decision signal. The test results, at a 10MHz clock, indicate that recognizing a single UA target requires 13.3 ns, with a power consumption of 189.02 mW. Subsequently, a more intensive test was performed by continuously feeding 1000 randomly selected UA target signals under 0 dB SNR conditions into the circuit. After analyzing all output results, the circuit demonstrated an accuracy of 95% in the recognition task of 1000 UA target signals at 0 dB SNR. It shows the effectiveness and robustness of the circuit on this UA target recognition task and dataset. Figure 10 shows the circuit layout for the UA target recognition convolutional chip, based on TSMC 65nm technology. Simulation results indicate that the total area of the neural network chip is 0.21 mm^2. The total power consumption is measured at 90.82 mW, with the recognition time for a single target being 7.81 ns. The ASIC implementation of the UA target recognition system exhibits a significant reduction in power consumption and a substantial increase in computational speed compared to previous designs. This improvement underscores the benefits of using advanced ASIC technology in enhancing the efficiency and performance of specialized neural network applications in acoustic recognition tasks.

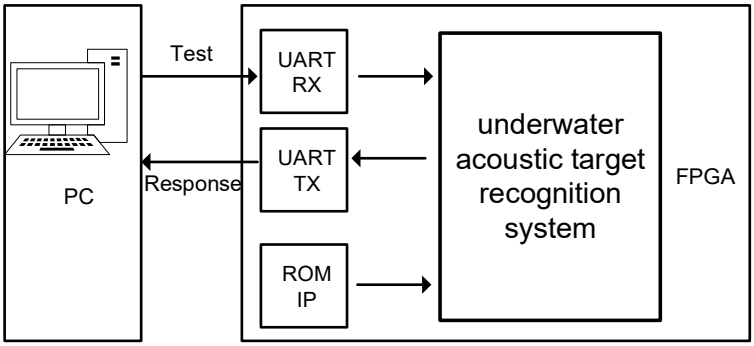


Figure 10. FPGA test system.

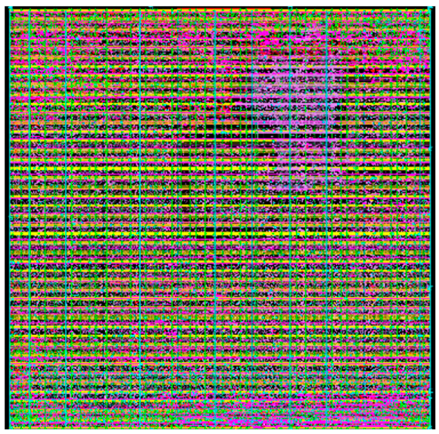


Figure 11. The layout of the UA target recognition ASIC chip.[1]

The physical layout of the ASIC chip, as shown in Figure 11, reflects the architectural design principles outlined in Figure 12. The design is organized to optimize data flow and minimize signal travel distance. The storage components, including caches for weights and feature maps, are strategically placed adjacent to the computational units to facilitate high-speed data access. The computational module, which includes the sparse matrix convolution unit and the accumulator, is designed as a highly parallel structure. This parallel architecture is visibly represented by the repetitive patterns of logic units across the chip layout. The placement of these units and their interconnects is optimized to support the dot product operations and subsequent aggregation, ensuring that the final convolution output is generated with maximum efficiency and minimal latency. This carefully designed physical implementation is key to achieving the chip's high performance and energy efficiency metrics.

The overall architecture of the system, including the storage and computational modules that contribute to its high energy efficiency, is presented in Figure 12. This design facilitates the impressive performance metrics achieved on the ASIC platform.

[1] This layout corresponds to the architecture shown in Figure 12. Different colored blocks represent various logic units and memory arrays, while the dense network of orthogonal lines signifies the metal interconnects that ensure high-speed data transfer between modules.

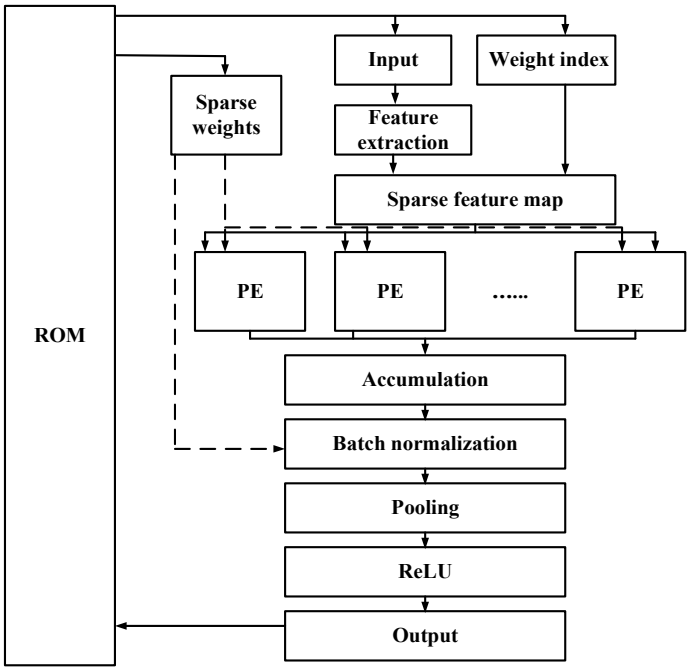


Figure 12. Architecture of the UA target recognition system.

Table 4 presents a performance comparison of the DenseNet convolutional neural network for underwater acoustic target recognition deployed across different platforms. The ASIC solution significantly reduces power consumption and greatly enhances computational speed for the same convolutional neural network operations. This comparison highlights the efficiency of the ASIC architecture in optimizing both energy usage and processing time for complex neural computations.

Table 4. Performance comparison on different platforms.

Implement ation	Platform	Device	Sparsity	Accuracy	Time	Power	Note
Software	CPU	13900K	0%	98.73%	--	--	Dense model
Software	GPU	RTX4090	0%	98.73%	--	--	Dense modell
Software	GPU	RTX4090	50%	96.11%	--	--	Pruned model
Hardware	FPGA	CYCLONE	50%	95.00%	13.33ns	189.02mW	Quantized
Hardware	ASIC	Custom	50%	95.00%	7.81ns	90.82mW	Quantized

Table 4 presents a comprehensive performance comparison across different implementation stages and platforms. The results demonstrate the progression from theoretical performance to practical hardware deployment:

Software implementations (CPU and GPU) with the dense model achieve identical accuracy of 98.73%, confirming the consistency of our algorithm across different computing platforms. The GPU implementation with 50% pruning shows 96.11% accuracy, representing the software performance after network sparsification.

Hardware implementations (FPGA and ASIC) both achieve 95% accuracy with the pruned model. The slight performance degradation from software (96.11%) to hardware (95%) is attributed to quantization effects and fixed-point arithmetic constraints inherent in hardware implementations. This 1.11% accuracy trade-off enables significant improvements in power consumption (90.82mW for ASIC vs 189.02mW for FPGA) and processing speed (7.81ns for ASIC vs 13.33ns for FPGA).

The ASIC implementation demonstrates the effectiveness of custom silicon design, achieving 52% power reduction and 41% speed improvement compared to FPGA while maintaining the same recognition accuracy.

4.3 Alternative Classification Methods

To comprehensively validate the effectiveness of our integrated softmax classifier approach, we conducted comparative experiments using traditional machine learning classifiers applied to features extracted from our DenseNet model. This analysis aims to demonstrate whether the end-to-end deep learning approach provides advantages over conventional feature extraction followed by separate classification.

Table 5. Performance comparison of different classification methods at 0dB SNR.

Classification Method	Feature Extraction	Accuracy (%)	Precision	Recall	F1-Score	Training Time	Inference Time
SVM (RBF kernel)	DenseNet features	94.2	0.941	0.938	0.939	45.2 min	2.3 ms
Random Forest	DenseNet features	92.8	0.926	0.925	0.925	12.8 min	0.8 ms
k-NN (k=5)	DenseNet features	89.3	0.891	0.887	0.889	--	15.6 ms
Naive Bayes	DenseNet features	85.7	0.854	0.851	0.852	2.1 min	0.5 ms
Softmax (Proposed)	End-to-end	98.73	0.987	0.987	0.987	180 min	0.05 ms

For this comparison, we extracted 512-dimensional feature vectors from the penultimate layer (before the softmax layer) of our trained DenseNet model. These features were then used to train various classical machine learning classifiers on the same 70%-30% train-test split of the ShipsEar dataset. All traditional classifiers were implemented using Python's scikit-learn library with optimized hyperparameters determined through 5-fold cross-validation.

Table X presents the comprehensive performance comparison of different classification approaches. The results reveal several important insights:

Support Vector Machine (SVM): Using an RBF kernel with optimal hyperparameters (C=100, γ=0.001), the SVM classifier achieved 94.2% accuracy. While this represents strong performance for a traditional approach, it falls 4.53 percentage points short of our integrated method. The SVM showed particularly good precision (0.941) but required significantly longer training time (45.2 minutes) and inference time (2.3 ms per sample).

Random Forest: This ensemble method achieved 92.8% accuracy with 100 trees and maximum depth of 20. Random Forest demonstrated consistent performance across all metrics with balanced precision and recall values around 0.925. The training time was notably shorter (12.8 minutes) compared to SVM, making it computationally efficient among traditional methods.

k-Nearest Neighbors (k-NN): With k=5 determined through cross-validation, k-NN achieved 89.3% accuracy. However, it suffered from the highest inference time (15.6 ms) due to the need to compute distances to all training samples, making it impractical for real-time applications.

Naive Bayes: This probabilistic classifier showed the lowest accuracy (85.7%) but the fastest training (2.1 minutes) and competitive inference time (0.5 ms). The assumption of feature independence likely contributed to its suboptimal performance on the complex acoustic features.

End-to-End Softmax (Proposed): Our integrated approach significantly outperformed all traditional methods with 98.73% accuracy and superior performance across all metrics. Despite requiring longer training time (180 minutes for the entire network), the inference time was remarkably fast (0.05 ms), demonstrating the efficiency of the optimized neural network architecture.

The superior performance of the integrated softmax approach can be attributed to several factors:

1. Feature Learning: Unlike traditional methods that rely on pre-extracted features, our approach learns optimal feature representations specifically for the classification task through end-to-end training.

2. Non-linear Decision Boundaries: The deep network architecture can capture complex non-linear relationships in the acoustic data that traditional linear or kernel-based methods may miss.

3. Task-Specific Optimization: The entire network, from feature extraction to classification, is jointly optimized for the underwater acoustic recognition task, ensuring coherent learning across all components.

4. Computational Efficiency: Once trained, the neural network provides the fastest inference time, making it suitable for real-time underwater applications.

These results confirm that the integrated deep learning approach not only achieves superior accuracy but also provides optimal computational efficiency for deployment in underwater acoustic target recognition systems. The significant performance gap (4.53% improvement over the best traditional method) justifies the adoption of the end-to-end learning paradigm for this challenging application domain.

5. Conclusion

This paper proposed the design of a high-energy-efficiency underwater acoustic target recognition system based on a sparse convolutional neural network circuit, specifically employing a pruned DenseNet model. To address issues related to network complexity, fine-grained pruning was utilized to sparsify the weights and reduce computational demands. Through comparative simulations, our DenseNet-based approach demonstrated superior performance, achieving the highest recognition accuracy of 98.73% at 0 dB SNR—2.84% higher than ResNet. Furthermore, we investigated the impact of different feature extraction methods and architectural parameters, determining that 1024-point FFT preprocessing and 5×5 convolutional kernels provided an optimal balance between accuracy and computational efficiency.

Software simulations show that at 0 dB SNR, the unpruned model's recognition accuracy reached 98.73%. After applying 50% fine-grained pruning, accuracy experienced a slight decrease to 96.11%, still validating the effectiveness and accuracy of the proposed method for this task. A prototype circuit was implemented on an FPGA, showing that recognition of a single UA target could be achieved in 13.3 ns with a power consumption of 189.02mW. When tested continuously on 1000 UA targets, the recognition accuracy reached 95%, indicating that the accelerator was capable of performing the recognition task effectively while maintaining high accuracy and efficiency.

Furthermore, the ASIC implementation demonstrated even better performance with a total chip area of 0.21 mm^2. The overall power consumption was reduced to 90.82mW, and the recognition time per target was significantly decreased to only 7.81 ns. This highlights a substantial improvement in both power efficiency and computational speed compared to the FPGA implementation and traditional software approaches, making the proposed system particularly suitable for energy-constrained underwater applications. The comprehensive experimental results across different platforms and conditions confirm the robustness and practicality of our high-energy-efficiency underwater acoustic target recognition system.

6. Limitations and Future Work

Despite these promising results, this study has certain limitations. First, the model was trained and evaluated primarily on the ShipsEar dataset, which exhibits significant class imbalance—such as the "Trawler" category comprising only about 1% of the samples. This may affect the generalizability and fairness of the model across all target categories. Second, the pruning strategy, while effective in reducing computational load, led to a noticeable decline in accuracy under low SNR conditions, indicating a trade-off between model sparsity and robustness in noisy environments. Future work should explore more adaptive pruning mechanisms and noise-robust training strategies to maintain performance across diverse acoustic conditions. Additionally, the current hardware implementations (FPGA and ASIC) were validated under laboratory settings; their performance in real-world underwater scenarios, with time-varying noise and environmental dynamics, remains to be extensively tested.

Author Contributions: Conceptualization, Wenhao Yang and Ding Ding; Data curation, Yinghao Lei, Liqin Zhu and Bingyao Peng; Formal analysis, Pei Tan; Funding acquisition, Ding Ding; Investigation, Bingyao Peng; Methodology, Wenhao Yang and Ding Ding; Project administration, Ding Ding; Resources, Liqin Zhu and Bingyao Peng; Software, Ao Ma, Pei Tan and Yinghao Lei; Supervision, Ding Ding; Validation, Ao Ma, Pei Tan and Yinghao Lei; Visualization, Liqin Zhu; Writing – original draft, Wenhao Yang and Ao Ma; Writing – review & editing, Ao Ma. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by 2025 National College Student Innovation Training Program Project, grant number 202510554085, and 2024 Hunan Provincial Undergraduate Innovation Training Program Project, grant number S202410554129, and The APC was funded by Hunan University of Technology and Business.

Data Availability Statement: Restrictions apply to the availability of these data. Data were obtained from the ShipsEar Dataset and are available at "https://underwaternoise.atlantic.uvigo.es/" with permission from the ShipsEar Dataset.

Acknowledgments: The authors would like to thank all individuals and institutions that provided support which is not covered by the author contribution or funding sections, including administrative and technical assistance.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
ASIC	Application-Specific Integrated Circuit
CNN	Convolutional Neural Network
CMOS	Complementary Metal-Oxide-Semiconductor
CSB	Compressed Sparse Banks
DNN	Deep Neural Networks
DT	Decision Trees
eLU	Exponential Linear Unit
FPGA	Field-Programmable Gate Arrays
GFCC	Gammatone Frequency Cepstral Coefficients
GMM	Gaussian Mixture Model
KNN	k-nearest neighbors
LSTM	Long Short-Term Memory
MEMD	Modified Empirical Mode Decomposition
ML	Machine Learning
ReLU	Rectified Linear Unit
SNR	Signal-to-Noise Ratio
SVM	Support Vector Machines
UA	Underwater Acoustic
UASN	Underwater Acoustic Sensor Networks

References

1. Wang, Y.; Jin, Y.; Zhang, H.; Lu, Q.; Cao, C.; Sang, Z.; Sun, M. Underwater communication signal recognition using sequence convolutional network. *IEEE Access* 2021, 9, 46886–46899.
2. Huang, J.; Diamant, R. Adaptive modulation for long-range underwater acoustic communication. *IEEE Trans. Wirel. Commun.* 2020, 19, 6844–6857.
3. Wang, L.; Yang, Y.; Liu, X. A direct position determination approach for underwater acoustic sensor networks. *IEEE Trans. Veh. Technol.* 2020, 69, 13033–13044.
4. Wang, X.; Liu, A.; Zhang, Y.; Xue, F. Underwater acoustic target recognition: A combination of multi-dimensional fusion features and modified deep neural network. *Remote Sens.* 2019, 11, 1888.

5. Zhang, C.; Yu, S.; Li, G.; Xu, Y. The recognition method of MQAM signals based on BP neural network and bird swarm algorithm. *IEEE Access* 2021, 9, 36078–36086.
6. Zeng, X.; Wang, S. Bark-wavelet analysis and Hilbert–Huang transform for underwater target recognition. *Def. Technol.* 2013, 9, 115–120.
7. Zhang, W.; Wu, Y.; Wang, D.; Wang, Y.; Wang, Y.; Zhang, L. Underwater target feature extraction and classification based on gammatone filter and machine learning. In *Proceedings of the 2018 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR),* Chengdu, China, 15–18 July 2018; IEEE: New York, NY, USA, 2018; pp. 42–47.
8. Ke, X.; Yuan, F.; Cheng, E. Underwater acoustic target recognition based on supervised feature-separation algorithm. *Sensors* 2018, 18, 4318.
9. Huang, G.; Liu, Z.; van der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
10. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
11. Cao, S.; Zhang, C.; Yao, Z.; Xiao, W.; Nie, L.; Zhan, D.; Liu, Y.; Wu, M.; Zhang, L. Efficient and effective sparse LSTM on FPGA with bank-balanced sparsity. In *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA '19)*, Seaside, CA, USA, 24–26 February 2019; pp. 63–72.
12. Chen, X.; Xu, G.; Xu, X.; Jiang, H.; Tian, Z.; Ma, T. Multicenter Hierarchical Federated Learning with Fault-Tolerance Mechanisms for Resilient Edge Computing Networks. *IEEE Trans. Neural Netw. Learn. Syst.* 2024, 36, 47–61.
13. Liang, W.; Chen, X.; Huang, S.; Xiong, G.; Yan, K.; Zhou, X. Federal Learning Edge Network Based Sentiment Analysis Combating Global COVID-19. *Comput. Commun.* 2023, 204, 33–42.
14. Mao, X.; Shan, Y.; Li, F.; Chen, X.; Zhang, S. CLSpell: Contrastive Learning with Phonological and Visual Knowledge for Chinese Spelling Check. *Neurocomputing* 2023, 554, 126468.
15. Fei, F.; Li, S.; Dai, H.; Hu, C.; Dou, W.; Ni, Q. A K-Anonymity Based Schema for Location Privacy Preservation. *IEEE Trans. Sustain. Comput.* 2017, 4, 156–167.
16. Zhou, X.; Li, Y.; Liang, W. CNN-RNN Based Intelligent Recommendation for Online Medical Pre-Diagnosis Support. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 2021, 18, 912–921.
17. Zhou, X.; Liang, W.; Wang, K.I.-K.; Wang, H.; Yang, L.T.; Jin, Q. Deep-Learning-Enhanced Human Activity Recognition for Internet of Healthcare Things. *IEEE Internet Things J.* 2020, 7, 6429–6438.
18. Zhou, X.; Xu, X.; Liang, W.; Zeng, Z.; Yan, Z. Deep-Learning-Enhanced Multitarget Detection for End–Edge–Cloud Surveillance in Smart IoT. *IEEE Internet Things J.* 2021, 8, 12588–12596.
19. Zhou, X.; Liang, W.; Li, W.; Yan, K.; Shimizu, S.; Wang, K.I.-K. Hierarchical Adversarial Attacks Against Graph-Neural-Network-Based IoT Network Intrusion Detection System. *IEEE Internet Things J.* 2021, 9, 9310–9319.
20. Peng, X.; Ren, J.; She, L.; Zhang, D.; Li, J.; Zhang, Y. BOAT: A Block-Streaming App Execution Scheme for Lightweight IoT Devices. *IEEE Internet Things J.* 2018, 5, 1816–1829.
21. Zhang, D.; Qiao, Y.; She, L.; Shen, R.; Ren, J.; Zhang, Y. Two Time-Scale Resource Management for Green Internet of Things Networks. *IEEE Internet Things J.* 2018, 6, 545–556.
22. Jiang, F.; Wang, K.; Dong, L.; Pan, C.; Xu, W.; Yang, K. AI Driven Heterogeneous MEC System with UAV Assistance for Dynamic Environment: Challenges and Solutions. *IEEE Netw.* 2020, 35, 400–408.
23. Wang, J.; Lv, P.; Wang, H.; Shi, C. SAR-U-Net: Squeeze-and-Excitation Block and Atrous Spatial Pyramid Pooling Based Residual U-Net for Automatic Liver Segmentation in Computed Tomography. *Comput. Methods Programs Biomed*. 2021, 208, 106268.
24. Zhou, X.; Liang, W.; Yan, K.; Li, W.; Wang, K. I.-K.; Ma, J.; Jin, Q. Edge-Enabled Two-Stage Scheduling Based on Deep Reinforcement Learning for Internet of Everything. *IEEE Internet Things J.* 2022, 10, 3295–3304.

25. Zhao, J.; Nguyen, H.; Nguyen-Thoi, T.; Asteris, P. G.; Zhou, J. Improved Levenberg–Marquardt Backpropagation Neural Network by Particle Swarm and Whale Optimization Algorithms to Predict the Deflection of RC Beams. *Eng. Comput.* 2022, 38 (Suppl. 5), 3847–3869.
26. Jiang, F.; Wang, K.; Dong, L.; Pan, C.; Yang, K. Stacked Autoencoder-Based Deep Reinforcement Learning for Online Resource Scheduling in Large-Scale MEC Networks. *IEEE Internet Things J.* 2020, 7, 9278–9290.
27. Xu, C.; Ren, J.; She, L.; Zhang, Y.; Qin, Z.; Ren, K. EdgeSanitizer: Locally Differentially Private Deep Inference at the Edge for Mobile Data Analytics. *IEEE Internet Things J.* 2019, 6, 5140–5151.
28. Jiang, F.; Dong, L.; Wang, K.; Yang, K.; Pan, C. Distributed Resource Scheduling for Large-Scale MEC Systems: A Multiagent Ensemble Deep Reinforcement Learning with Imitation Acceleration. *IEEE Internet Things J.* 2021, 9, 6597–6610.
29. Ouyang, Y.; Liu, W.; Yang, Q.; Mao, X.; Li, F. Trust Based Task Offloading Scheme in UAV-Enhanced Edge Computing Network. *Peer-to-Peer Netw. Appl.* 2021, 14, 3268–3290.
30. Zhang, J.; Bhuiyan, M. Z. A.; Yang, X.; Wang, T.; Xu, X.; Hayajneh, T.; Khan, F. AntiConcealer: Reliable Detection of Adversary Concealed Behaviors in EdgeAI-Assisted IoT. *IEEE Internet Things J.* 2021, 9, 22184–22193.
31. Zhang, P.; Wang, J.; Guo, K. Compressive Sensing and Random Walk Based Data Collection in Wireless Sensor Networks. *Comput. Commun.* 2018, 129, 43–53.
32. Zhao, Y.; Xie, G.; Chen, H.; Chen, M.; Huang, L. Enhancing Underwater Acoustic Target Recognition Through Advanced Feature Fusion and Deep Learning. *J. Mar. Sci. Eng.* 2025, *13*, 278.
33. Yao, H.; Gao, T.; Wang, Y.; Wang, H.; Chen, X. Mobile_ViT: Underwater Acoustic Target Recognition Method Based on Local–Global Feature Fusion. *J. Mar. Sci. Eng.* 2024, 12, 589.
34. Chen, Z.; Tang, J.; Qiu, H.; Chen, M. MGFGNet: An Automatic Underwater Acoustic Target Recognition Method Based on the Multi-Gradient Flow Global Feature Enhancement Network. *Frontiers in Marine Science*. 2023, 10, 1306229.
35. Wang, Y.; Zhang, H.; Huang, W.; Zhou, M.; Gao, Y.; An, Y.; Jiao, H. DWSTr: A Hybrid Framework for Ship-Radiated Noise Recognition. *Frontiers in Marine Science* 2024, 11, 1334057.
36. Yang, K.; Wang, B.; Fang, Z.; Cai, B. An End-to-End Underwater Acoustic Target Recognition Model Based on One-Dimensional Convolution and Transformer. *J. Mar. Sci. Eng.* 2024, *12*, 1793.