International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

Development of CNN-LSTM Model for Object Detection and Classification in Sonar Imagery in Maritime Defense Applications

Alan Fhajoeng Ramadhan[1], Gentio Harsono[2], Achmad Farid Wadjdi[3]

[1]Department of Sensing Technology, Faculty of Defense Science and Technology
Republic of Indonesia Defense University
Jakarta, Indonesia
alan.ramadhan@tp.idu.ac.id
[2]Department of Sensing Technology, Faculty of Defense Science and Technology
Republic of Indonesia Defense University
Jakarta, Indonesia
hgentio1969@gmail.com
[3]Research Center for Artificial Intelligence and Cyber Security
National Research and Innovation Agency
Jakarta, Indonesia
zahramdr@gmail.com

Abstract— This research develops a combination model of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) for the identification and classification of objects in sonar images, applied in the context of maritime defense and security. CNN is used to extract spatial features from sonar images, while LSTM processes the temporal sequences of these features to enhance the accuracy of underwater object classification. The dataset used consists of eight object classes: shipwrecks, plane wrecks, drowning victims, bottles, propellers, submarines, mines, and tires. The model was trained and tested using normalized and augmented data to enhance the variation and quality of the training data. The evaluation results show that the CNN-LSTM model achieves high accuracy in classifying underwater objects. At the end of the training, the training and validation accuracy reached 100% after 100 epochs, demonstrating the model's excellent ability to generalize knowledge from training data to unseen data. Additionally, the consistently decreasing loss value during the training process indicates the model's effectiveness in reducing prediction errors. This research proves that the combination of CNN and LSTM is an effective approach for identifying and classifying underwater objects in sonar images. With these promising results, the CNN-LSTM model has the potential to be implemented in real-world applications, supporting efforts to detect and identify underwater objects quickly and accurately, and contributing to the enhancement of maritime safety and security. This research makes a significant contribution to the development of underwater object detection and identification technology, which is crucial for maritime defense and security.

Keywords— Convolutional Neural Network, Long Short-Term Memory, Sonar Imaging, Object Identification, Object Classification, Maritime Defense, Maritime Security.

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

I. INTRODUCTION

Maritime defense and security are crucial elements in maintaining the sovereignty of maritime areas and protecting strategic assets in the waters. Threats from underwater objects such as mines [1] [2] [3], submarines [4] [5], and marine debris [6] [7] necessitate a reliable detection and identification system. Sonar technology, which uses sound waves for navigation [8], communication [9], and detecting underwater objects [10], is highly relevant in this context. However, the main challenge is how to identify and classify objects with high accuracy in various complex marine environmental conditions.

The use of sonar images in the identification of underwater objects has become standard in military and civilian operations. However, the interpretation of complex sonar images is often influenced by various factors such as noise [11], image resolution [12], and underwater environmental conditions [13]. Deep learning, particularly CNN and LSTM, offers solutions to enhance the accuracy of underwater object identification and classification by optimizing the processes of spatial and temporal feature extraction and analysis.

This research aims to develop a program for object identification and classification in sonar images, specifically applied in maritime defense and security. The proposed approach combines Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) to leverage the strengths of each model in extracting spatial and temporal features from sonar images. CNN is one of the Deep Learning (DL) models used in image analysis for tasks such as image classification, segmentation, detection, registration, and content-based image retrieval [14]. Applying CNN in processing sonar images can be achieved using an active learning-based algorithm that seeks informative images from unlabeled data and continuously retrains the model [15]. The use of CNN in processing sonar images can improve accuracy in object matching in deep ocean environments with dynamic backgrounds, high intensity, and high noise levels [16]. One of the advantages of CNN is its ability to extract features from images with high accuracy [17]. Therefore, in this study, the CNN model is used to extract important features from sonar images.

Next, the feature extraction results from the CNN model are then processed with the LSTM model to handle the temporal sequences of these features to improve classification accuracy. LSTM is a Deep Recurrent Neural Network architecture used to extract dependencies from sequential data [18], such as time-series data classification [19] using an LSTM encoder-decoder model to reconstruct the input sequence [20]. This model is also capable of capturing temporal dependencies and evolution patterns in dynamic networks [21], resulting in more accurate predictions and visualizations of image data [22]. The selection of the LSTM model is due to its ability to model complex evolution patterns behind spatiotemporal time series. Whereas traditional feedforward neural networks cannot understand the complex interactions occurring between various locations in different time series [23].

Advanced underwater object identification and classification systems can identify potential threats earlier [1], allowing for faster and more effective preventive measures. In addition, this system can also support search and rescue operations for drowning victims [24] as well as other civil applications that require accurate underwater monitoring. In the long term, the development of this technology will contribute to the enhancement of maritime safety and security as well as support efforts to maintain the sovereignty of territorial waters. Therefore, using DL models such as the combination of CNN-LSTM in this research is expected to enhance the capability of underwater object detection, utilize temporal information to improve classification accuracy, and be implementable in real-world applications for maritime defense and security. This research holds high significance in the context of maritime defense and security.

Research method

This research develops a program for object identification and classification in sonar images for maritime defense and security using a combination of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM). This model is designed to extract spatial features from sonar images through CNN and process the temporal sequences of these features with LSTM to enhance classification accuracy.

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

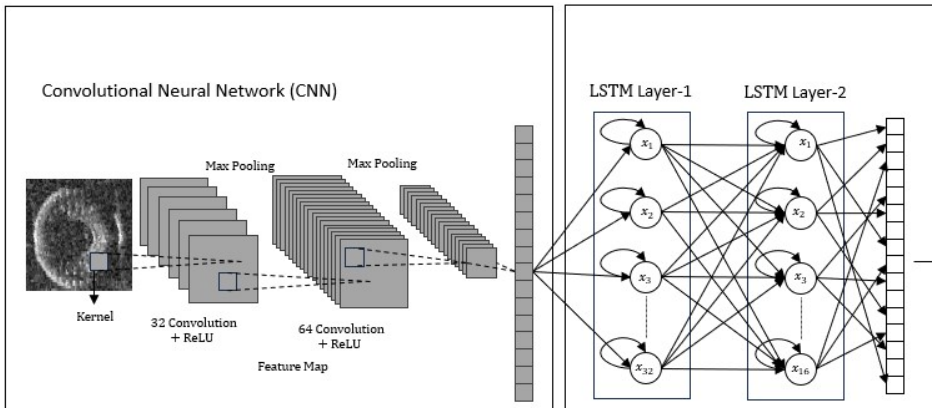

Fig 1. CNN-LSTM Model Architecture

A. Data Collection

The dataset used in this research consists of sonar images containing various underwater objects such as mines, submarines, and other sea debris. This data was collected from various sources that provide sonar images in RGB format with a size of 128x128 pixels. The training data used consists of eight object classes: shipwrecks, plane wrecks, drowning victims, bottles, propellers, submarines, mines, and tires. The entire dataset is divided into training data (70%) and test data (30%).

B. Data Pre-processing

The preprocessing stage includes normalization and data augmentation. Normalization is performed by changing pixel values to the range [0, 1] to accelerate model convergence, reduce computational resources, and improve accuracy [25]. Data augmentation is carried out to increase the variation of training data and includes operations such as rotation, flipping, and image translation [26]

C. Model Architecture

The developed model consists of two main components: CNN for extracting spatial features and LSTM for processing the temporal sequence of those features The model architecture is shown in Fig 1.

1) Input Layer

The input layer receives sonar images with a size of 128x128 pixels in RGB format, which means each image has three color channels. (merah, hijau, biru). Input shape for this model is (128,128, 3).

2) First Convolutional Layer

The convolutional layer in the CNN architecture helps automatically learn features from images using training data and high computational resources such as GPUs [27]. The first convolutional layer aims to extract basic features from sonar images. In this layer, 32 convolutional filters with a size of 3x3 and the ReLU activation function are used. The padding used is "same," which means the output has the same size as the input.

- Convolutional Operation

$$(F * I)(x,y) = \sum_{i-0}^{k-1}\sum_{j=0}^{k-1} F(i,j) \cdot I(X+i, y+j) \quad (1)$$

Where F is the convolution filter, I is the input image, k is the filter size, and (x,y) is the position in the output image.

- ReLU Activation Function

$$ReLU(x) = \max(0, x) \tag{2}$$

This function is used to introduce non-linearity into the model [28].

3) First Pooling Layer

The pooling layer is a spatial down-sampling layer that gradually reduces the feature map, increases the receptive field size, and decreases the number of parameters in the model [29]. The first pooling layer serves to reduce the spatial dimensions of the image, thereby reducing the number of parameters and computations in the network. In this layer, max pooling operation with a size of 2x2 is used. Max Pooling Operation:

$$P(x, y) = \max_{(i,j) \epsilon R} I(x + i, y + j) \tag{3}$$

Where R is the Pooling area and P is the pooling result.

4) Second Convolutional Layer

The second convolutional layer aims to extract more complex features from the sonar image. In this layer, 64 convolutional filters with a size of 3x3 and the ReLU activation function are used. The padding used is also "same."

- Convolutional Operation

$$(F * I)(x, y) = \sum_{i-0}^{k-1} \sum_{j=0}^{k-1} F(i, j) \cdot I(X + i, y + j) \tag{4}$$

- ReLU Activation Function

$$ReLU(x) = \max(0, x) \tag{5}$$

5) Second Pooling Layer

The second pooling layer is used to further reduce the spatial dimensions of the image. A max pooling operation with a size of 2x2 is also applied in this layer. Max Pooling Operation:

$$P(x, y) = \max_{(i,j) \epsilon R} I(x + i, y + j) \tag{6}$$

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

6) Flatten Layer

The Flattened layer converts the output from the second pooling layer, which is in the form of a two-dimensional matrix, into a one-dimensional vector. This is necessary so that the data can be input into the LSTM layer. The Flatten operation is as follows:

$$Flatten(x) = [x_1, x_2, \dots, x_n] \tag{7}$$

7) Repeat Vector Layer

The Repeat Vector layer repeats the input vector (Flatten) so that it can be fed into the LSTM layer as a temporal sequence. For example, if we want to have 10 timesteps, the input vector will be repeated 10 times.

$$ReperatVector(x) = [x, x, \dots, x]\ (10\ times) \tag{8}$$

8) First LSTM Layer

The first LSTM layer is used to process the temporal sequence of features extracted by the CNN. This layer consists of 32 units with return_sequences=True, which means the output from each timestep will be passed to the next timestep. Each timestep in this sequence is processed by the LSTM units shown in Fig. 2.

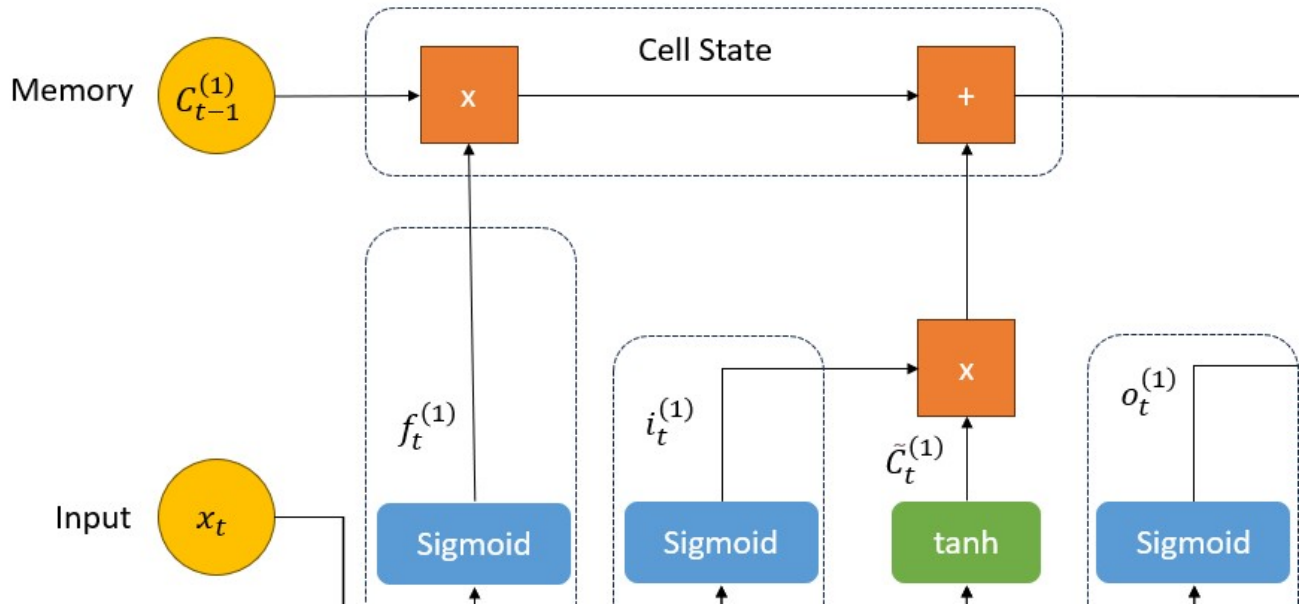

Fig. 2 First LSTM Layer

At each timestep *t*, the LSTM receives input data x_t and the hidden state $h_{t-1}^{(1)}$ from the previous timestep. Along with that, the previous cell state $C_{t-1}^{(1)}$ is also included. The first process that occurs is the calculation of the input gate $(i_t^{(1)})$,

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

which determines how much new information will be stored in the cell state. This input gate uses weights W_xi^((1)) and $W_{hi}^{(1)}$ as well as bias $b_i^{(1)}$, and the result is activated by the sigmoid function.

$$i_t^{(1)} = \sigma(W_{xi}^{(1)} x_t + W_{hi}^{(1)} h_{t-1}^{(1)} + b_i^{(1)}) \tag{9}$$

Next, the forget gate $(f_t^{(1)})$ is calculated to determine how much information from the previous cell state $C_{t-1}^{(1)}$ will be discarded. Like the input gate, the forget gate uses weights $W_{xf}^{(1)}$and $W_{hf}^{(1)}$as well as bias $b_f^{(1)}$, and is activated by the sigmoid function.

$$f_t^{(1)} = \sigma(W_{xf}^{(1)} x_t + W_{hf}^{(1)} h_{t-1}^{(1)} + b_f^{(1)}) \tag{10}$$

Then, the candidate cell state $(\tilde{C}_t^{(1)})$ is calculated. This is the new value proposed to be added to the current cell state. Candidate cell state is calculated using weights $W_{xC}^{(1)}$and $W_{hC}^{(1)}$as well as bias $b_C^{(1)}$, and the result is activated by the tanh function to produce values between -1 and 1.

$$\tilde{C}_t^{(1)} = tanh\left(W_{xC}^{(1)} x_t + W_{hC}^{(1)} h_{t-1}^{(1)} + b_C^{(1)}\right) \tag{11}$$

With information from the forget gate and candidate cell state, the current cell state $C_t^{(1)}$ is updated. The forget gate $f_t^{(1)}$ determines the portion of the previous cell state $C_{t-1}^{(1)}$ that is retained, while the input gate $i_t^{(1)}$ regulates how much new information from the candidate cell state $C_t^{(1)}$ is added to the current cell state. The result is the updated cell state $C_t^{(1)}$.

$$C_t^{(1)} = f_t^{(1)} \cdot C_{t-1}^{(1)} + i_t^{(1)} \cdot \tilde{C}_t^{(1)} \tag{12}$$

Next, the output gate $(o_t^{(1)})$ is calculated to determine how much information from the current cell state will be produced as the hidden state. This output gate uses weights $W_{xo}^{(1)}$ and $W_{ho}^{(1)}$as well as bias $b_o^{(1)}$, and is activated by the sigmoid function.

$$o_t^{(1)} = \sigma(W_{xo}^{(1)} x_t + W_{ho}^{(1)} h_{t-1}^{(1)} + b_o^{(1)}) \tag{13}$$

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

Finally, the hidden state $(h_t^{(1)})$ is calculated by combining the current cell state $C_t^{(1)}$ modulated by the output gate o_t through the tanh function. This hidden state $h_t^{(1)}$ is then used as input for the next timestep or as the final output of the network.

$$h_t^{(1)} = o_t^{(1)} . \tanh(C_t^{(1)}) \tag{14}$$

9) Second LSTM Layer

The second LSTM layer consists of 16 units and does not use return_sequences=True, which means only the output from the last timestep will be passed to the next layer. The second LSTM layer receives the output from the first LSTM layer as input. The process is similar to the first LSTM layer, but with different parameters.

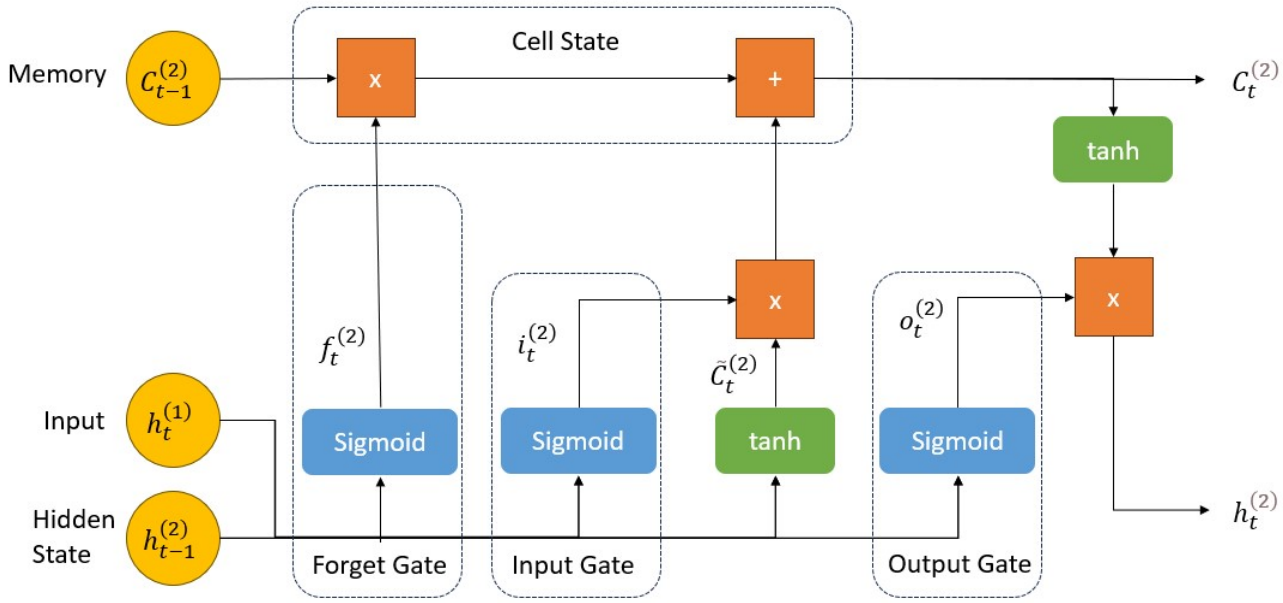

Fig. 3 Second LSTM Layer

At each timestep *t*, LSTM Layer 2 receives input from the hidden state produced by First LSTM Layer $h_t^{(1)}$ and the hidden state from the previous timestep on LSTM Layer 2 $h_{t-1}^{(2)}$. Along with that, the previous cell state $C_{t-1}^{(2)}$ is also included. The first process that occurs is the calculation of the input gate $i_t^{(2)}$, which determines how much new information from the hidden state of LSTM Layer 1 will be stored in the cell state. This input gate uses weights and $W_{xi}^{(2)}$ and $W_{hi}^{(2)}$ as well as bias $b_i^{(2)}$, and the result is activated by the sigmoid function.

$$i_t^{(2)} = \sigma(W_{xi}^{(2)} x_t + W_{hi}^{(2)} h_{t-1}^{(2)} + b_i^{(2)}) \tag{15}$$

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

Next, the forget gate $(f_t^{(2)})$ is calculated to determine how much information from the previous cell state $C_{t-1}^{(2)}$ will be discarded. Like the input gate, the forget gate uses weights $W_{xf}^{(2)}$ and $W_{hf}^{(2)}$ as well as bias $b_f^{(2)}$, and is activated by the sigmoid function.

$$f_t^{(2)} = \sigma(W_{xf}^{(2)} h_t^{(1)} + W_{hf}^{(2)} h_{t-1}^{(2)} + b_f^{(2)}) \tag{16}$$

Then, the candidate cell state $(\tilde{C}_t^{(2)})$ is calculated. This is the new value proposed to be added to the current cell state. Candidate cell state is calculated using weights $W_{xC}^{(2)}$ and $W_{hC}^{(2)}$ as well as bias $b_C^{(2)}$, and the result is activated by the tanh function to produce values between -1 and 1.

$$\tilde{C}_t^{(2)} = tanh\left(W_{xC}^{(2)} h_t^{(1)} + W_{hC}^{(2)} h_{t-1}^{(2)} + b_C^{(2)}\right) \tag{17}$$

With information from the forget gate and candidate cell state, the current cell state $C_t^{(2)}$ is updated. The forget gate $f_t^{(2)}$ determines the portion of the previous cell state $(C_{t-1}^{(2)})$ that is retained, while the input gate $i_t^{(2)}$ regulates how much new information from the candidate cell state $\tilde{C}_t^{(2)}$ is added to the current cell state. The result is the updated cell state $C_t^{(2)}$.

$$C_t^{(2)} = f_t^{(2)} \cdot C_{t-1}^{(2)} + i_t^{(2)} \cdot \tilde{C}_t^{(2)} \tag{18}$$

Next, the output gate $(o_t^{(2)})$ is calculated to determine how much information from the current cell state will be produced as the hidden state. This output gate uses weights $W_{xo}^{(2)}$ and $W_{ho}^{(2)}$ as well as bias $b_o^{(2)}$, and is activated by the sigmoid function.

$$o_t^{(2)} = \sigma(W_{xo}^{(2)} h_t^{(1)} + W_{ho}^{(2)} h_{t-1}^{(2)} + b_o^{(2)}) \tag{19}$$

Finally, the hidden state $(h_t^{(2)})$ is calculated by combining the current cell state $C_t^{(2)}$ modulated by the output gate $o_t^{(2)}$ through the tanh function. This hidden state $h_t^{(2)}$ is then used as input for the next timestep or as the final output of the network.

$$h_t^{(2)} = o_t^{(2)} . \tanh(C_t^{(2)}) \tag{20}$$

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

10) Output Layer

The output from the second LSTM layer is fed into the Dense layer with a softmax activation function to produce classification predictions:

$$Dense = softmax(W_d \cdot h_t^{(2)} + b_d) \tag{21}$$

$$softmax(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{22}$$

Where W_d is the weight and b_d is the bias for the Dense layer, and $h_t^{(2)}$ is the hidden state from the last timestep in the second LSTM layer, z_i is the i-th element of the input vector z. The softmax activation function transforms the output into probabilities that sum to 1 [30].

D. Model Training

The model was trained using the backpropagation through time (BPTT) algorithm with the Adam optimizer [31]. The loss function used is categorical cross-entropy [32], which is defined as:

$$\mathcal{L} = -\sum_i y_i \log(\hat{y}_i) \tag{23}$$

Where y_i is the actual value of class i. $\hat{y}_i$ is the predicted probability of class i.

II. Result and Discussion

The model developed in this study has been evaluated using a sonar image dataset consisting of eight object classes: shipwreck, plane wreck, drowning victim, bottle, propeller, submarine, mine, and tire. This model was evaluated based on accuracy and loss metrics on the test data. Here are the evaluation results of the proposed CNN-LSTM model.

A. Model Performance

The proposed model consists of several layers, namely the convolutional layer, pooling layer, flatten layer, repeat vector layer, and LSTM layer. Each layer has a specific role in the feature extraction and object classification process. Here is a detailed analysis of the performance of each layer:

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

TABLE I. MODEL SUMMARY

Layer (type)	Output Shape	Param
Convolution 1 (Conv2D)	(None, 128, 128, 32)	896
Max Pooling (MaxPooling2D)	(None, 64, 64, 32)	0
Convolution 2 (Conv2D)	(None, 64, 64, 64)	18496
Max Pooling (MaxPooling2D)	(None, 32, 32, 64)	0
Flatten (Flatten)	(None, 65536)	0
Repeat_Vector (RepeatVector)	(None, 10, 65536)	0
LSTM 1 (LSTM)	(None, 10, 32)	8392832
LSTM 2 (LSTM	(None, 16)	3136
Dense (Dense)	(None, 8)	136

Total Param : 25246490

Trainable Param : 8415496

Non-trainable Param : 0

Optimizer Param : 16830994

B. Accuracy Result

The obtained accuracy data includes training and validation accuracy measured at each step or epoch during the training process. This dataset shows how the model learns over time and its ability to generalize knowledge to previously unseen data.

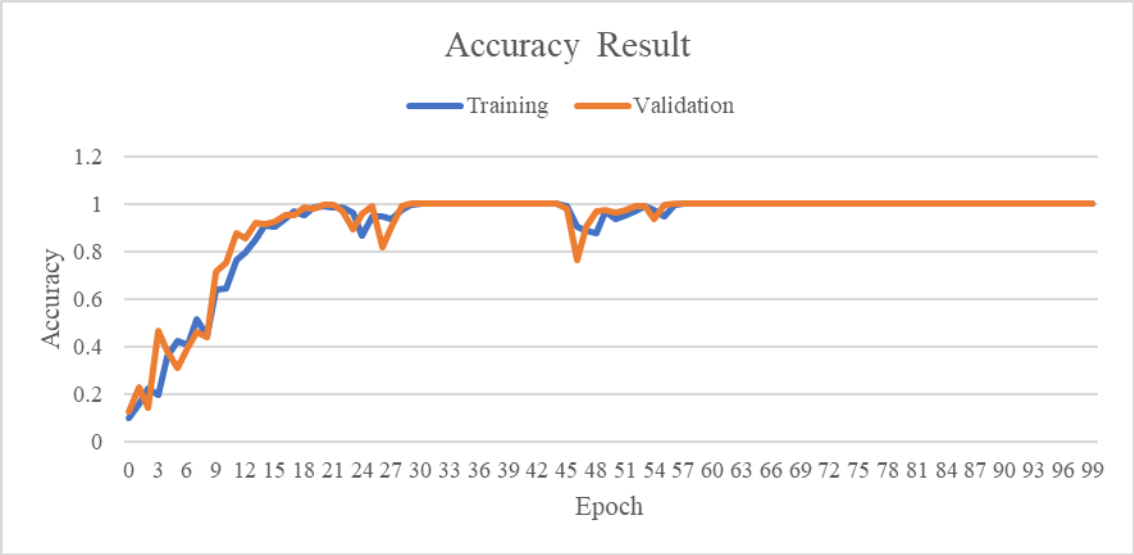

Fig 4. Accuracy Result

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

Fig 4. shows at the beginning of the training, the training accuracy started at a value of 0.100. This shows that the model initially could only make correct predictions for 10% of the total training data. As the epochs increase, the training accuracy begins to gradually improve. At epoch 2, the training accuracy rose to 0.16. The model continued to learn and began to capture more information from the training data until it reached 0.985 at epoch 23. However, there was a slight decrease in training accuracy from epoch 24 to 30 and from epoch 46 to 56, which could be caused by fluctuations in the learning process. The training accuracy significantly increased to 1.00 and stabilized from epoch 56 to 100, indicating an improvement in the model's ability to predict the correct class.

The validation accuracy at the beginning of the training is also low, starting at a value of 0.125. This indicates that the model initially had poor performance in generalizing knowledge to the validation data. Validation accuracy provides an indication of how well the model can generalize its knowledge to new data that is not included in the training data. Similar to training, at epoch 2, the validation accuracy increased to 0.23. The model began to learn to generalize its knowledge well, reaching 0.995 by epoch 22. However, there was a slight decrease in training accuracy from epoch 22 to 30 and from epoch 46 to 56, which could be due to the model possibly having difficulty generalizing knowledge, but the values were still high compared to previous epochs, indicating stability in learning. Fluctuations in validation accuracy indicate that the model is still in the adjustment phase and is seeking a balance between overfitting and underfitting. Validation accuracy increased significantly to 1.00 and stabilized from epoch 57 to 100. Despite the fluctuations, validation accuracy shows a stable upward trend, which is a positive indication of the model's ability to generalize knowledge.

C. Loss Result

The data loss obtained includes the loss on the training and validation data measured at each step or epoch during the training process. This dataset shows how the model learns and reduces its prediction errors over time.

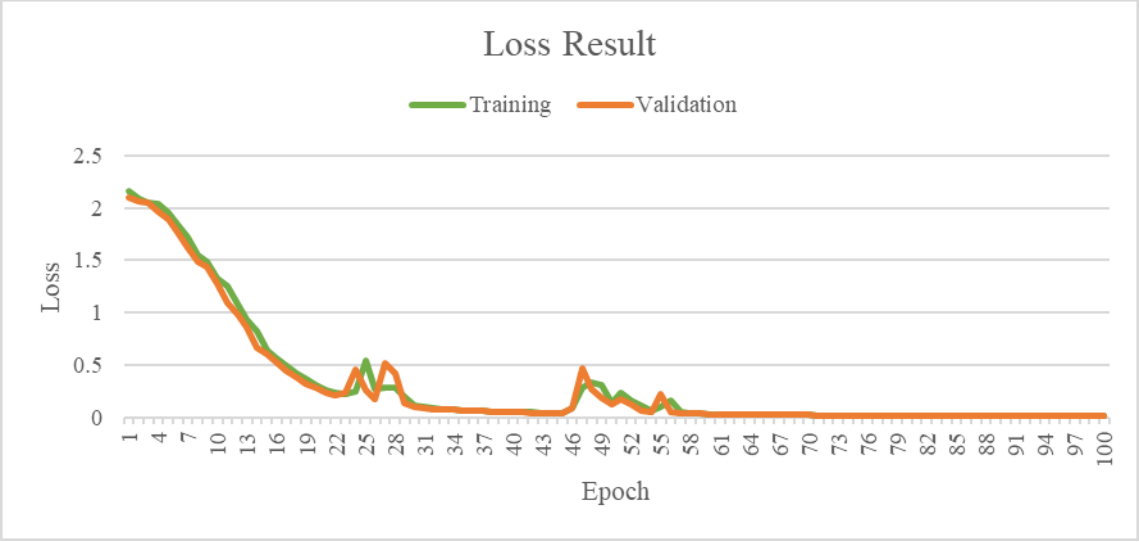

Fig 5. Loss Result

Fig. 5 shows at the beginning of the training, the training loss value started at 2.155397, indicating that the model initially had a very high prediction error. However, as the epochs increased, the training loss value gradually decreased, showing that the model was learning and adjusting its weights to reduce prediction errors. At the second epoch, the training loss value decreased to 2.092412. This decrease indicates that the model is starting to learn the basic patterns from the training data and reducing its prediction errors. At epoch 6, the training loss further decreased to 1.9542, indicating that the model is becoming better at capturing

International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

https://ijpsat.org/
Vol. 48 No. 1 December 2024, pp. 102-116

information from the training data. By epoch 23, the training loss gradually decreased to 0.227261, showing stability in the model's learning process. At epoch 64, the training loss experienced a significant decrease to 0.029755. This decrease indicates an improvement in the model's ability to predict more accurately, substantially reducing prediction errors. Overall, the decrease in training loss values indicates that the model is able to learn well from the training data, improve its predictions, and adjust its weights to minimize errors. The consistent decrease in training loss values indicates that the model is on the right track in the learning process.

Meanwhile, the validation loss value indicates how well the model can generalize its knowledge to new data that is not included in the training data. At the beginning of the training, the validation loss started at 2.104337, indicating that the model's prediction error on unseen data was also very high. In the second epoch, the validation loss value decreased to 2.062312. This decrease indicates that the model is beginning to learn to generalize its knowledge, although it is still in the early stages. In the third epoch, the validation loss slightly decreased to 2.052337, indicating that the model is still in the process of learning to improve its generalization ability. In the third epoch, the validation loss significantly decreased to 1.962931. This is a positive sign that the model is starting to learn to generalize well, capturing important patterns in the validation data that are similar to the training data. At the 70th epoch, the validation loss further decreased to 0.020384, indicating stability in learning and the model's ability to generalize its knowledge better. The stable decrease in validation loss indicates that the model is capable of learning from the training data and applying the acquired knowledge to make more accurate predictions on new data. This suggests that the model is not only memorizing the training data but also understanding patterns that can be applied to previously unseen data.

D. Model Test Result

The results of the combination model of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) shown in this analysis reflect significant advancements in object recognition and classification in the context of sequential data, such as videos or image sequences. This model leverages the power of CNN in feature extraction from images, which is known to be very effective in recognizing visual patterns [33], as well as the capability of LSTM in handling sequential data by storing temporal information [34]. This combination creates a system that can not only recognize objects in static images but also understand the surrounding temporal context, thereby enabling more accurate predictions in dynamic situations.

In the obtained results, the model in Fig. 6 shows very good performance with high accuracy across various object categories. For example, categories such as "Aircraft Wreckage" and "Drowning Victims" recorded an accuracy of over 99%, indicating that this model is capable of recognizing and classifying objects with a very high level of reliability. This is very important in real-world applications, such as in search and rescue, where quick and accurate detection can save lives. Additionally, other categories like "Bottle" and "Tire" also show very good results, with accuracy above 99%. This indicates that the model has been trained with sufficiently representative and diverse data, allowing the system to capture the important features of each object well.

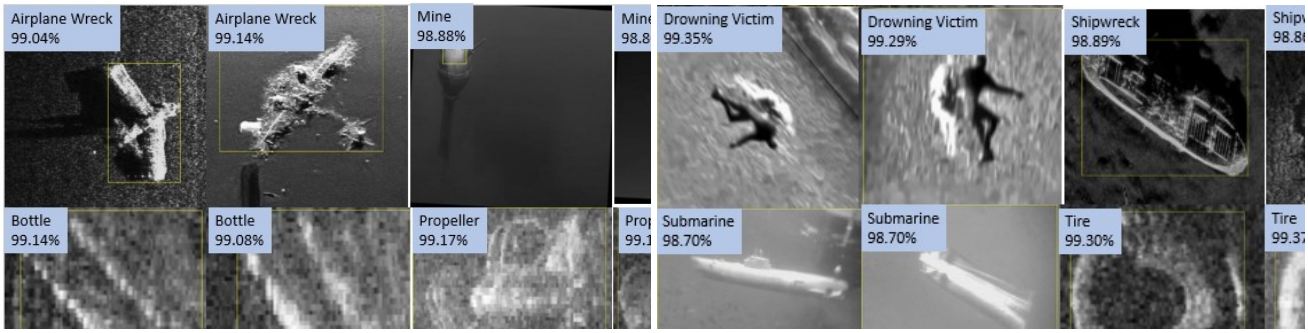

Fig. 6. Model Test Result

However, despite the impressive results obtained, there are several categories that show slightly lower accuracy, such as "Shipwreck" and "Submarine," each achieving an accuracy of around 98%. This decline may be caused by several factors, such as the visual complexity of these objects, where the distinguishing features between categories may not be strong enough to allow the

model to classify more accurately. Additionally, variation in the training data also plays a crucial role; if the data used to train the model is not diverse or representative enough, the model may not be able to learn well to recognize objects under different conditions.

From an application perspective, the combination of CNN and LSTM opens up great opportunities for various practical uses. In the field of security monitoring, for example, this system can be used to detect and identify objects in videos in real-time, enhancing surveillance efficiency. In the environmental sector, this model can assist in monitoring potentially hazardous objects, such as hazardous waste or invasive species, by providing more accurate and timely information. With these promising results, the next steps in this research could include further development of the training dataset to improve accuracy in more challenging categories, as well as the exploration of new techniques in data augmentation to enrich the variety of images used. Thus, these results not only demonstrate the remarkable potential of the CNN and LSTM combination model but also provide a solid foundation for the development of more advanced and effective object recognition technologies in the future.

III. CONCLUSION

This research has successfully developed and evaluated a combination model of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) for the identification and classification of objects in sonar images, specifically applied in the context of maritime defense and security. This model is designed to extract spatial features from sonar images through CNN and process the temporal sequences of these features with LSTM, aiming to improve the accuracy of underwater object classification in various complex environmental conditions. The evaluation results show that the CNN-LSTM model is capable of achieving high accuracy in classifying eight classes of underwater objects, including shipwrecks, plane wrecks, drowning victims, bottles, propellers, submarines, mines, and tires. At the end of the training, this model showed a stable training accuracy of 100%, and a validation accuracy that also reached 100% after 100 epochs. This indicates that the model has an excellent ability to generalize the knowledge obtained from the training data to previously unseen data. Additionally, the consistently decreasing loss value during the training process shows that the model is capable of effectively reducing its prediction errors.

Overall, this research proves that the combination of CNN and LSTM is an effective approach for the identification and classification of underwater objects in sonar images. This model is not only capable of extracting important features from sonar images but also utilizes temporal information to enhance classification accuracy. With these promising results, the CNN-LSTM model can be implemented in real-world applications for maritime defense and security, supporting efforts to detect and identify underwater objects more quickly and accurately, ultimately contributing to the improvement of maritime safety and security.

ACKNOWLEDGMENT

As the author, I want to sincerely thank all of the instructors and tutors at Indonesian Defense University's Remote Sensing Technology Study Program. Special thanks go out to the second and third authors, Mr. Gentio Harsono and Mr. Achmad Farid Wadjdi, for their advice and direction in carrying out this study. I would like to express my gratitude to the Navy's Hydro-Oceanography Center in for their cooperation in providing data for this work, as well as for their advice and insight. We also appreciate the help and advice provided by the paper's reviewers.

REFERENCES

[1] N. Palomeras, T. Furfaro, D. P. Williams, M. Carreras, and S. Dugelay, “Automatic Target Recognition for Mine Countermeasure Missions Using Forward-Looking Sonar Data,” *IEEE Journal of Oceanic Engineering*, vol. 47, no. 1, 2022, doi: 10.1109/JOE.2021.3103269.

[2] E. Dura, Y. Zhang, X. Liao, G. J. Dobeck, and L. Carin, “Active learning for detection of mine-like objects in side-scan sonar imagery,” *IEEE Journal of Oceanic Engineering*, vol. 30, no. 2, 2005, doi: 10.1109/JOE.2005.850931.

[3] S. Hożyń, “A review of underwater mine detection and classification in sonar imagery,” 2021. doi: 10.3390/electronics10232943.

[4] D. Eleftherakis and R. Vicen, "Sensors to increase the security of underwater communication cables: A review of underwater monitoring sensors," *Sensors (Switzerland)*, vol. 20, no. 3, 2020, doi: 10.3390/s20030737.

[5] W. Shin, D. S. Kim, and H. Ko, "Target Tracking from Weak Acoustic Signals in an Underwater Environment Using a Deep Segmentation Network," *J Mar Sci Eng*, vol. 11, no. 8, 2023, doi: 10.3390/jmse11081584.

[6] J. Watanabe, Y. Shao, and N. Miura, "Underwater and airborne monitoring of marine ecosystems and debris," *J Appl Remote Sens*, vol. 13, no. 04, 2019, doi: 10.1117/1.jrs.13.044509.

[7] A. Preciado-Grijalva, B. Wehbe, M. B. Firvida, and M. Valdenegro-Toro, "Self-supervised Learning for Sonar Image Classification," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2022. doi: 10.1109/CVPRW56347.2022.00156.

[8] A. Miller, B. Miller, and G. Miller, "Navigation of underwater drones and integration of acoustic sensing with onboard inertial navigation system," 2021. doi: 10.3390/drones5030083.

[9] Y. Wang, Z. Shi, X. Ma, and L. Liu, "A Joint Sonar-Communication System Based on Multicarrier Waveforms," *IEEE Signal Process Lett*, vol. 29, 2022, doi: 10.1109/LSP.2021.3106889.

[10] U. Anitha, S. Malarkkan, G. D. Anbarasi Jebaselvi, and R. Narmadha, "Sonar image segmentation and quality assessment using prominent image processing techniques," *Applied Acoustics*, vol. 148, 2019, doi: 10.1016/j.apacoust.2018.12.038.

[11] F. Yuan, F. Xiao, K. Zhang, Y. Huang, and E. Cheng, "Noise reduction for sonar images by statistical analysis and fields of experts," *J Vis Commun Image Represent*, vol. 74, 2021, doi: 10.1016/j.jvcir.2020.102995.

[12] R. Kumudham and V. Rajendran, "Super resolution enhancement of underwater sonar images," *SN Appl Sci*, vol. 1, no. 8, 2019, doi: 10.1007/s42452-019-0886-5.

[13] A. P. Lyons, D. R. Olson, and R. E. Hansen, "Modeling the effect of random roughness on synthetic aperture sonar image statistics," *J Acoust Soc Am*, vol. 152, no. 3, 2022, doi: 10.1121/10.0013837.

[14] H. Yu, L. T. Yang, Q. Zhang, D. Armstrong, and M. J. Deen, "Convolutional neural networks for medical image analysis: State-of-the-art, comparisons, improvement and perspectives," *Neurocomputing*, vol. 444, 2021, doi: 10.1016/j.neucom.2020.04.157.

[15] L. Jiang, T. Cai, Q. Ma, F. Xu, and S. Wang, "Active Object Detection in Sonar Images," *IEEE Access*, vol. 8, 2020, doi: 10.1109/ACCESS.2020.2999341.

[16] W. Yang, S. Fan, S. Xu, P. King, B. Kang, and E. Kim, "Autonomous Underwater Vehicle Navigation Using Sonar Image Matching based on Convolutional Neural Network," in *IFAC-PapersOnLine*, 2019. doi: 10.1016/j.ifacol.2019.12.300.

[17] A. Yang, X. Yang, W. Wu, H. Liu, and Y. Zhuansun, "Research on feature extraction of tumor image based on convolutional neural network," *IEEE Access*, vol. 7, 2019, doi: 10.1109/ACCESS.2019.2897131.

[18] A. Sepas-Moghaddam, A. Etemad, F. Pereira, and P. L. Correia, "Long Short-Term Memory with Gate and State Level Fusion for Light Field-Based Face Recognition," *IEEE Transactions on Information Forensics and Security*, vol. 16, 2021, doi: 10.1109/TIFS.2020.3036242.

[19] T. D. Pham, "Time–frequency time–space LSTM for robust classification of physiological signals," *Sci Rep*, vol. 11, no. 1, 2021, doi: 10.1038/s41598-021-86432-7.

[20] S. Saha, F. Bovolo, and L. Bruzzone, "Change Detection in Image Time-Series Using Unsupervised LSTM," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, 2022, doi: 10.1109/LGRS.2020.3043822.

[21] M. Yang, J. Liu, L. Chen, Z. Zhao, X. Chen, and Y. Shen, "An Advanced Deep Generative Framework for Temporal Link Prediction in Dynamic Networks," *IEEE Trans Cybern*, vol. 50, no. 12, 2020, doi: 10.1109/TCYB.2019.2920268.

[22] W. Fang, F. Zhang, Y. Ding, and J. Sheng, "A new sequential image prediction method based on LSTM and DCGAN," *Computers, Materials and Continua*, vol. 64, no. 1, 2020, doi: 10.32604/CMC.2020.06395.

[23] X. Xu and M. Yoneda, "Multitask Air-Quality Prediction Based on LSTM-Autoencoder Model," *IEEE Trans Cybern*, vol. 51, no. 5, 2021, doi: 10.1109/TCYB.2019.2945999.

[24] G. Huo, Z. Wu, and J. Li, "Underwater Object Classification in Sidescan Sonar Images Using Deep Transfer Learning and Semisynthetic Training Data," *IEEE Access*, vol. 8, 2020, doi: 10.1109/ACCESS.2020.2978880.

[25] H. A. Monteiro, A. V. de Brito, and E. U. K. Melcker, "Image normalization in embedded systems," *J Real Time Image Process*, vol. 18, no. 6, 2021, doi: 10.1007/s11554-021-01098-8.

[26] K. Meethongjan, V. T. Hoang, and T. Surinwarangkoon, "Data augmentation by combining feature selection and color features for image classification," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 6, 2022, doi: 10.11591/ijece.v12i6.pp6172-6177.

[27] M. A. Saleem, N. Senan, F. Wahid, M. Aamir, A. Samad, and M. Khan, "Comparative Analysis of Recent Architecture of Convolutional Neural Network," *Math Probl Eng*, vol. 2022, 2022, doi: 10.1155/2022/7313612.

[28] Y. Ying, J. Su, P. Shan, L. Miao, X. Wang, and S. Peng, "Rectified Exponential Units for Convolutional Neural Networks," *IEEE Access*, vol. 7, 2019, doi: 10.1109/ACCESS.2019.2928442.

[29] M. A. Mahmoudi, A. Chetouani, F. Boufera, and H. Tabia, "Learnable pooling weights for facial expression recognition," *Pattern Recognit Lett*, vol. 138, 2020, doi: 10.1016/j.patrec.2020.09.001.

[30] S. Mehra, G. Raut, R. Das Purkayastha, S. K. Vishvakarma, and A. Biasizzo, "An Empirical Evaluation of Enhanced Performance Softmax Function in Deep Learning," *IEEE Access*, vol. 11, 2023, doi: 10.1109/ACCESS.2023.3265327.

[31] F. Mehmood, S. Ahmad, and T. K. Whangbo, "An Efficient Optimization Technique for Training Deep Neural Networks," *Mathematics*, vol. 11, no. 6, 2023, doi: 10.3390/math11061360.

[32] A. Rusiecki, "Trimmed categorical cross-entropy for deep learning with label noise," *Electron Lett*, vol. 55, no. 6, 2019, doi: 10.1049/el.2018.7980.

[33] Q. Yao, Y. Wang, and Y. Yang, "Underwater Acoustic Target Recognition Based on Data Augmentation and Residual CNN," *Electronics (Switzerland)*, vol. 12, no. 5, Mar. 2023, doi: 10.3390/electronics12051206.

[34] H. A. N. Huimei, Z. H. U. Xingquan, and L. I. Ying, "Generalizing long short-term memory network for deep learning from generic data," *ACM Trans Knowl Discov Data*, vol. 14, no. 2, 2020, doi: 10.1145/3366022.